






Article

Investigation of Microservice-Based Workflow Management Solutions for Industrial Automation

Jaime Garcia Represa ¹, Felix Larrinaga ², Pal Varga ^{3,*}, William Ochoa ², Alain Perez ², Dániel Kozma ³
and Jerker Delsing ¹

¹ Cyber-Physical Systems EISLAB, SRT, Lulea University of Technology, 97187 Luleå, Sweden

² Information Systems, Faculty of Engineering, Mondragon Unibertsitatea, Arrasate-Mondragon, 20500 Arrasate, Gipuzkoa, Spain

³ Department of Telecommunications and Media Informatics, Faculty of Electrical Engineering and Informatics, Budapest University of Technology and Economics, Műegyetem rkp. 3., H-1111 Budapest, Hungary

* Correspondence: pvarga@tmit.bme.hu

Abstract: In an era ruled by data and information, engineers need new tools to cope with the increased complexity of industrial operations. New architectural models for industry enable open communication environments, where workflows can play a major role in providing flexible and dynamic interactions between systems. Workflows help engineers maintain precise control over their factory equipment and Information Technology (IT) services, from the initial design stages to plant operations. The current application of workflows departs from the classic business workflows that focus on office automation systems in favor of a manufacturing-oriented approach that involves direct interaction with cyber-physical systems (CPSs) on the shop floor. This paper identifies relevant industry-related challenges that hinder the adoption of workflow technology, which are classified within the context of a cohesive workflow lifecycle. The classification compares the various workflow management solutions and systems used to monitor and execute workflows. These solutions have been developed alongside the Eclipse Arrowhead framework, which provides a common infrastructure for designing systems according to the microservice architectural principles. This paper investigates and compares various solutions for workflow management and execution in light of the associated industrial requirements. Further, it compares various microservice-based approaches and their implementation. The objective is to support industrial stakeholders in their decision-making with regard to choosing among workflow management solutions.

Keywords: Arrowhead framework; business process management; workflow management; workflow execution; service-oriented architecture; microservices; web service automation



Citation: Represa, J.G.; Larrinaga, F.; Varga, P.; Ochoa, W.; Perez, A.; Kozma, D.; Delsing, J. Investigation of Microservice-Based Workflow Management Solutions for Industrial Automation. *Appl. Sci.* **2023**, *13*, 1835. <https://doi.org/10.3390/app13031835>

Academic Editor: Antonella Petrillo

Received: 16 December 2022

Revised: 25 January 2023

Accepted: 26 January 2023

Published: 31 January 2023



Copyright: © 2023 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

The anticipated Fourth Industrial Revolution remains a long-term goal for most companies, even though the current market conditions urge a paradigm change. There is a technological and organizational gap that has to be addressed before the industry can perform the transition to Industry 4.0. From Cyber-Physical Systems (CPSs) and the Internet of Things (IoT) to big data and artificial intelligence (AI), the technologies currently in development are paving the way for the digitalization of industry. However, new systems that incorporate these technologies need to work together and showcase the additional value that they can provide with respect to enhancing manufacturing. Otherwise, companies will not take the risk of modifying their already-working equipment and automation architectures.

While the current prevalent automation architecture ANSI/ISA-95 [1] meets expectations and has improved companies' competitiveness, it has predominantly resulted in designed time definitions for most variables. The result is factories with very stiff automation solutions that are based on a set of monolithic software tools glued together into a working solution by dedicated middleware. Changing these automation systems is

very expensive and time-consuming, often requiring the whole automation solution to be re-tested for even the smallest change in a workstation.

Upgrading legacy ISA-95 automation architectures and the associated technology to a microservice architecture has the potential to provide the required manufacturing flexibility. The upgrade involves more than just replacing manufacturing equipment with newer devices. There is still a big gap in communications between the model proposed in the ISA-95 standard and the new models that are aligned with the Industry 4.0 vision, such as the Reference Architectural Model for Industry 4.0 (RAMI 4.0) [2] or the Industrial Internet Reference Architecture (IIRA) [3]. Communications in factories that follow the ISA-95 hierarchical model are constrained in terms of the functional levels such that connections are only possible between elements at bordering levels. In contrast, new architectures enable and promote company-wide communications, where the shop floor and the business departments can exchange information directly without needing to route data through the automation levels, as shown in Figure 1. Forward-looking reference architectures for Industry 4.0 also expand upon the one-dimensional view of the ISA-95 architecture, incorporating other viewpoints central to industrial operations, such as the product and equipment lifecycle [4].

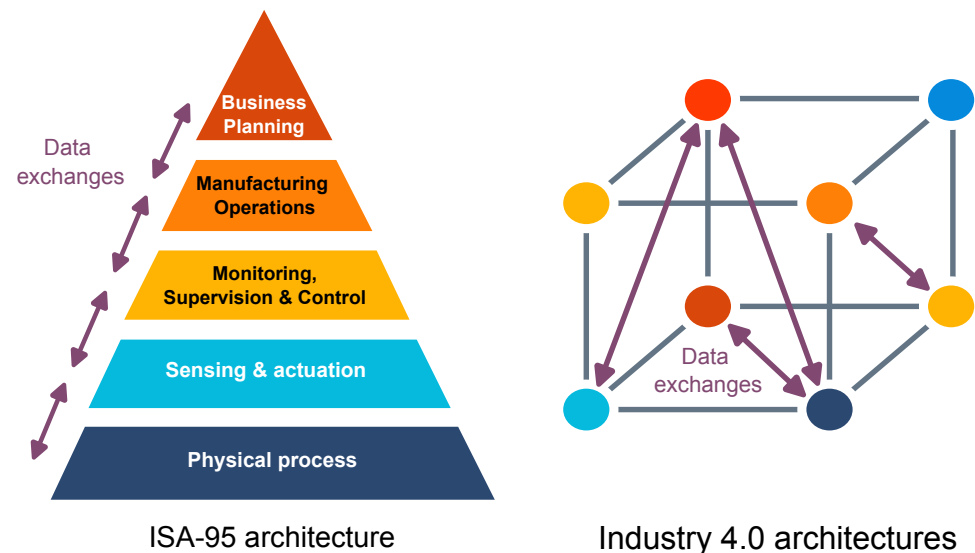


Figure 1. Hierarchical vs. meshed communication models in ISA-95 vs. Industry 4.0 architectures.

The relaxation of communication restrictions opens the door to new ways of organizing shop floor operations to fulfill new and dynamic market requirements. The requirements associated with dynamic production and Industry 4.0 introduce new demands with regard to production workflow management and execution [5]. Such dynamic requirements and demands cannot be addressed during production automation's design time. Thus, runtime reconfiguration and re-organization of, e.g., workflow management and execution, becomes important and achieving this calls for the usage of new technology.

This paper investigates microservice-based approaches with respect to their applicability to engineering design workflow management and execution. The workflow management disciplines described in this paper cover the issues of business process engineering, design process management, and manufacturing workflow management and primarily focus on microservice-based architectural solutions. Research surveys from this domain, such as [6,7], provide a good general overview of the field. However, the current paper is an investigative work in a specific automation engineering area associated with microservice approaches. While the usage of workflows is a mature discipline in the legacy ISA-95 realm, workflow implementation faces a new set of challenges in modern Industry 4.0 environments where distributed architectures, such as the service-oriented architecture (SOA) or the microservice architecture, are predominant. Thus, this work considers how

workflow management and execution based on a microservice architecture can meet the requirements of Industry 4.0 and dynamic markets.

For this purpose, the challenges and requirements related to industrial workflow technology were compiled from industry research project reports and scientific publications. The requirements aggregate was then used to evaluate the capabilities of a set of actively developed open source microservice-based workflow management and execution technologies.

Although this paper compares open source workflow management solutions that use microservice architectures, the approach used to identify the associated challenges and alternatives follows a particular methodology to conduct thorough studies. This methodology is outlined in [8] and follows the well-known PICOC (Population, Intervention, Comparison, Outcome, and Context) criteria for research question formulation. For this approach, and according to PICOC criteria, this study considered microservice architectures as the application area or domain (population), business process as the technology (intervention), workflow management as the specific technology for which the interventions are compared (comparison), and challenges in the industry as the factors of importance (outcome).

The main contributions of this paper are as follows:

1. It identifies the challenges for the industry with regard to choosing workflow management and execution solutions, which are grouped into a workflow lifecycle, and finds that microservice-based approaches address said challenges properly, especially those related to flexibility and dynamic data exchange.
2. It investigates different microservice-based approaches for workflow management and execution.
3. It investigates and compares how the various microservice-based workflow management methods address industrial requirements and challenges.
4. It compares the concrete implementation features of the microservice-based solutions to support decision-making in various scenarios.

This paper is structured as follows: Section 2 discusses the background and related work on workflows and microservices. In Section 3, the requirements and challenges compiled from previous research projects and the scientific literature are classified into a workflow lifecycle. Section 4 presents the set of solutions for workflow management, which are compared and matched to the requirements. Section 5 contains a discussion highlighting the differences identified between each approach studied. Finally, Section 6 concludes the paper by describing the main contributions of this research.

2. Background and Related Work

This section introduces workflows and microservice architectures, followed by a review of similar work in the field of Business Process Management (BPM) and Workflow Management Systems (WFMSs).

2.1. Workflows and Business Processes

Numerous devices are expected to communicate and share data with each other in modern production systems, as information-driven decisions are a vital advantage of Industry 4.0 organizations. Among the expected benefits that factory-wide communications will bring to companies are dynamic business processes and increased engineering flexibility [9]. In this scenario, WFMSs can support production by managing and organizing heterogeneous systems, their interactions, and their evolving behaviors.

The Workflow Management (WFM) and BPM fields were created during the rise of information systems, when companies were searching for new tools to coordinate a growing number of resources and automate complex processes. Although these fields have evolved, generally accepted definitions of their primary terms, business process, and workflow are still lacking in the scientific literature [10]. In the scope of this work, business processes are described as the manner in which an organization's resources are used to provide an output, adding value by fulfilling certain business goals. In an effort to develop industry standards in this domain, the Workflow Management Coalition (WfMC) [11] was

created, which defined workflows as the way business processes are automated according to a set of procedural rules. Another important contribution into the field was the workflow reference model (Figure 2), which has had a significant influence on the modular design of WFMSs for interoperability [12]. Nevertheless, the WfMC had decreased its activity in recent years, proceeding to its dissolution in 2019 [13], while development in the field is still ongoing.

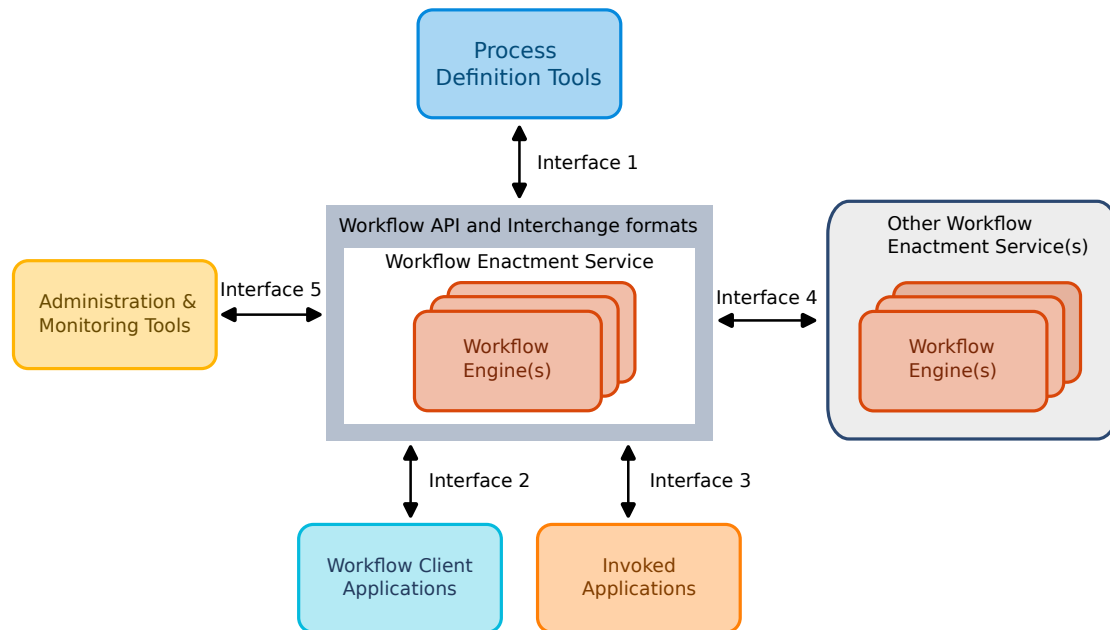


Figure 2. The workflow reference model, adapted from [11].

Of the different types of workflows in use today, this work explores the use of manufacturing workflows, which aim to be incorporated into the new generation of automation architectures. They are based upon business workflows and their emphasis on the control-flow perspective while focusing on coordinating software systems and relegating human tasks. Their advantage resides in the advanced control they offer over the factory shop floor operations by the execution of the workflow task through granular calls to machine services. The creation of manufacturing workflows would follow a Product Lifecycle Management (PLM) approach and be planned, scheduled, and executed by a Manufacturing Execution System (MES).

The lack of well-established definitions for the basic terminology of workflows and business processes reflects the absence of standardization in the field. From the workflow modeling languages to the WFMSs, their basic functionality and properties are not harmonized across the range of commercial offerings. Providers should consider their conformance with related solutions more thoroughly before creating their own offerings. Standards have been published, but, as argued in [14], they have failed to reach widespread adoption as they were not built upon a comprehensive workflow theory. Aside from the communication problems that engineers face when designing new WFMSs, the lack of standardization in the field is also represented by volatile requirements. There is a broad market of WFMSs that focus on particular niches, prompting practitioners to re-assess the requirements for each project as market demands shift.

2.2. Microservice Architectures

Microservice architecture originated from SOA, which was introduced by IBM in the mid-1990s [15]. Following the increase in popularity of SOA, the Organization for the Advancement of Structured Information Standards (OASIS) created the SOA reference model (SOA-RM) [16] to standardize its concepts in a common vocabulary. As SOA implementa-

tions matured and the term attracted increasing attention, its definition broadened, losing precision (<https://martinfowler.com/bliki/ServiceOrientedAmbiguity.html>, accessed on 10 December 2022). However, practitioners still required precision when communicating architectural terms and architectures evolved to overcome the complexity of certain SOA deployments. Eventually, a new term, “microservice architecture”, was coined as an evolution of the original concepts. Differences are highlighted in the communication between services—where microservice architectures promote lightweight communication protocols and simpler mechanisms for connecting services, not always relying on orchestration [17]. From a fundamental perspective, disregarding the scope of use, the two architectures can be considered equivalent for our purposes, as they share the basic principles of service orientation [18].

The pillar of microservice architecture is the encapsulation of functionality in services that can be provided to, or consumed by, other software systems. The design of services is considered the defining feature of service-based architectures. It enables independent software development unconstrained from the rest of the systems in the architecture, by having only a few dependencies among services. Additionally, software lifecycles are decoupled, as each service can evolve at its own pace without the coupling of legacy architectures, while service functionality is independent of any other software system, the degree to which the services are responsible for their own data can vary. There are two main design approaches to data persistence within services:

- Stateful services: Each service needs to store the communication state and keep track of the exchanges, reacting differently depending on previous requests.
- Stateless services: The services are not required to record the state of communications. Consumers will obtain the same response regardless of any previous exchanges between the systems.

In the case of stateful services, developers can proceed in two ways: (1) designing services to carry their own database, as in microservices, or (2) have a shared database that is accessible by the authorized services, which is more common in SOA [19].

Although microservice architectures are currently the main approach followed by companies looking to realize the automation and digitalization of their processes, there is no single correct methodology for such implementation, as each author is promoting their approach. Instead, there are several platforms that support organizations with the transition, providing the infrastructure for concrete microservice solutions, e.g., FIWARE [20], Eclipse BaSyx [21], and Eclipse Arrowhead [5,22]. In this work, the use of the Eclipse Arrowhead reference implementation is associated with the research projects funding this study and the analysis that links the essential properties of “look-up”, “late binding”, and “loose coupling”, shared by SOA and microservice architectures, to the mandatory systems of the framework [23]. A recent comparison of the most relevant initiatives related to automation and digitalization can be found in [24].

2.3. Services for Workflow Management

The subject of workflow management technologies precedes that of SOA and the more recent microservice architecture. Its origins can be traced back to the development of office automation in the late 1970s and 1980s [25]. Therefore, most research efforts have been undertaken from the point of view of integrating new software technologies with previously devised WFMSs.

The first interactions between the fields occurred in the early 2000s, when companies such as IBM explored how business process activities could be implemented via web services [26]. Their research showed how real-world activities could be captured through web services, decreasing the struggle of software integration in workflow systems that required ad hoc connections to monolithic software systems. Additionally, they claim that inter-company processes could be streamlined using technologies to compose web services. A similar approach has been followed to include newer resources, such as IoT devices, into the enterprise resource planning layer to be used in business processes [27]. More recent advancements in the use of services for implementing workflows fall under the umbrella

of Intelligent Business Process Management Suite (iBPMS) software [28]. The analysis in Gartner's "Magic Quadrant for Intelligent Business Process Management Suites (iBPMS) 2019" (<https://www.gartner.com/en/documents/3899484/>, accessed on 22 June 2021) shows offerings that do not provide industry domain capabilities by default.

Several articles have identified the need to change the industrial paradigm concerning business processes. For example, [29] recognizes that control information traditionally managed by an MES must be extended to provide accurate data exchanges to coordinate collaborative production processes. Further, [30] points out that a migration from traditional production systems to a modern approach is necessary. Traditional systems are usually rigid vertical applications using a centralized approach. On the other hand, modern approaches are agile plug-and-produce systems that are dynamically adaptable to changing production conditions, open to new features and functions, flexible to different processing tasks, modular to enable quick and economic changes, and able to support new business processes. In addition, [31] identifies that Industry 4.0 requires a multitude of data to reflect the demands of service-oriented manufacturing processes and describe the challenges of the orchestration process.

A search of the relevant scientific literature has identified articles that include adaptations in the industrial manufacturing context as a part of their proposal. For instance, [32] identifies the need to enable event-driven devices using microservices and the possibility of creating integrated workflows from data ingestion among the primary Industry 4.0 requirements. In [33], Barz M. et al. proposed a service-oriented architecture for integrating factory workers in industrial cyber-physical production environments; they used Camunda for modeling and execution. Similarly, in [34], Suri K. et al. created a semantic framework for developing IoT-aware business processes. Signavio (<https://www.signavio.com/>, accessed on 22 June 2021) was extended to support semantically annotated Business Process Modeling Notation (BPMN) in the process editor. [35] proposes a service-based framework architecture that uses Semantic Web technology and a workflow engine for service orchestration. Further, ref. [36] supposes a reference architecture and agile development method for engineering platforms via a Process Driven Approach (PDA) based on the BPMN standard and process engines. Moreover, [37] introduces a higher-level workflow and cloud-based SOA for individuals to design and prototype products and for manufacturing organizations to focus on their core competencies, identify complementary services that are a part of their production processes, and publish their idle capabilities. In addition, some current articles propose state-of-the-art technologies to support business processes in the industry, such as blockchain [38–40], Semantic Web technologies [41], or digital twins [42]. However, most of these industrial proposals lack a framework to support microservice architecture requirements as a whole.

Among the proposals that consider a framework to support microservice architectures, it is important to mention FITMAN-CBPM, which was created as a part of the deliverables of the European project "Future Internet Technologies for MANufacturing (FITMAN)" [43]. The framework provides a collaborative business process management (CBPM) component in the form of a web-based platform for the design, execution, and monitoring of semantically enhanced BPMN 2.0 business processes in service-oriented manufacturing ecosystems [44]. FITMAN-CBPM was created as an integral part of FIWARE's virtual factory reference architecture [45]. However, the platform is enormous and can be complex to implement without technical support [46]. Moreover, the Github repository (<https://github.com/CBPM-WG/Fitman-CBPM>, accessed on 10 December 2022) has not been updated since 2015, which was when the latest commit was written.

Overall, the literature reflects interest in the research of business processes for industry based on microservice architectures [47,48], but little implementation and validation through use cases are found, with only theoretical examples. In addition, the reviewed studies include several limitations. Either the systems are too complex to be implemented or they are simulations or research prototypes in their early stages, with a lack of support from a framework. Although extending already-existing open source platforms seems to

be a well-accepted procedure, without any validation or guarantees of their viability, it could be a doomed endeavor from the start. This is exacerbated by a lack of a supporting community behind most of the projects.

3. Challenges and Requirements Set by the Industry for the Workflow Lifecycle

Several reports have been published with valuable insights into the challenges faced by companies looking to adopt workflow technology [49], derived from the work in European research projects, Productive 4.0 (<https://cordis.europa.eu/project/id/737459>, accessed on 10 December 2022), and Arrowhead Tools (<https://cordis.europa.eu/project/id/826452>, accessed on 10 December 2022), where companies and research institutes have worked together. This information has been supplemented with a scientific literature survey of past WFM requirements [25,50]. Finally, the main barriers for industry have been grouped into four focus areas:

Architecture: From the architectural perspective, a company needs to decide how the WFMS is going to interact with the rest of the company's applications and devices. It needs to decide whether the steps of a workflow are controlled by a central system (orchestration) or else the control is passed at each step of the workflow to the next system executing it (choreography). A comparison of both approaches, orchestration vs. choreography, is presented in [51]. Another architectural constraint is the type of workflow considering the entity in charge of dispatching the model. The workflow can follow the traditional automation hierarchy and be PLM-based; the workflow recipe can be carried out by the products themselves through a smart product approach or can consider a hierarchical architecture that relies on a set of manufacturing equipment connected together that works as a production unit.

Engineering Life Cycle: Integrated into this area are a myriad of different lifecycles, from the overall project lifecycle to each individual shop floor equipment, which is used as resources when implementing the workflows. Although they will all have some impact on the suitability of the WFMS, it will be too complex to study them all. Therefore, a good way to start will be to approach the two most crucial phases of a general engineering process lifecycle when creating and managing workflows [52]. Design time workflow engineering encompasses the time spent designing the workflow, combining the functionality of different services in a specified order to achieve the desired business goal. Run-time workflow engineering, which refers to the period during which the workflow is executed in the deployed environment, corresponds to the operation & management step in the engineering process. Companies should consider how much dynamism is required in their workflows and at which stages.

Process–Task Relation: This area analyzes the characteristics of the workflows and their individual tasks. When considering the process-task relation, an organization must determine the aspects that are related to the workflow nature, such as parallel or sequential flow structure, nature of tasks (including human interaction or exclusively machine services), communication between tasks, interoperability, and the nature of services (reliability, dynamism, or concurrency).

Context: The new Industry 4.0 approach toward WFM also impacts the features that companies demand from new solutions. Thus, aspects such as scalability (replicability), security and privacy, collaborative process management (cross organization), and service management (service discovery) introduce new constraints, such as the addition of new infrastructure (message brokers, interoperability converters, firewalls, etc.).

The following subsections identify the individual requirements of WFMSs throughout the phases of the workflow lifecycle introduced in this paper. At each phase, those requirements that were deemed more relevant for the comparison of workflow solutions have been labeled with a challenge tag and used to analyze the WFMSs described in Section 4. Figure 3 depicts the workflow lifecycle phases and various internal challenges within those phases.



Figure 3. Workflow lifecycle phases and their internal challenges.

3.1. Requirements for Workflow Generation

Workflow generation is the phase of the lifecycle where the workflow is composed of the tasks required to achieve the desired output. Three main challenges are identified at this stage.

Challenge 1. Workflow modeling: Architectural design and workflow representation are the main concerns in this stage. Workflow modeling describes the logical model used to represent workflows and considers architectural constraints. Different languages are used to represent workflows and accommodate their design and architectural elements. According to [49], the requirements related to workflow representation are as follows:

- **Fit for collaborative context:** The workflow language must provide standardized syntax and semantics for workflows in collaborative contexts. This includes utilities to generate and manage workflows or represent partial workflows.
- **Fit for generation:** A workflow language must support the generation of workflows by providing components built with character-encoding standards without visual representation data. The language should offer these components in an order-independent manner that enables workflow composition.
- **Compactness:** A workflow language must represent the specifications of all the essential components and their attributes as well as the details of a workflow. Other components and details, such as information related to the graphical model display for humans or similar data, are optional.
- **Compositionality:** The workflow language must uniquely identify each workflow and enable referencing from other workflows. References to other workflows substitute actions built with components, thus providing references to sub-workflows.
- **Open semantics:** The workflow language must provide components that represent domain- and activity-specific information. The syntax of these elements is not defined by the workflow language syntax specification.

- **Extensibility:** The workflow language should be capable of being enhanced with constructs that extend the workflow syntax with optional components, elements, and values. For example, modules containing policies or quality of service definitions may be used to augment the syntax.

The sequence structure selected by a company conditions the language and tools to represent a workflow. Workflow recipes can be formalized into business processes that contain composable tasks [33]. The languages used to create formal workflows are BPMN, Business Process Execution Language (BPEL), state machines, petri nets, Yet Another Workflow Language (YAWL), etc. [53]. Moreover, the Object Management Group (OMG) ratified BPMN as the standard language for the design of business processes [54].

Challenge 2. Heterogeneous infrastructure scale [55]: This is an architectural challenge where designers must decide regarding the right level of aggregation/abstraction to compose workflows, considering the unit of execution decreases in edge environments, but interaction with the central cloud orchestrator systems is required. The architecture must support scalability by aggregating functionality in sub-workflows (e.g., using hierarchical models). At the same time, designers will require the abstract representation of units to cope with many similar devices performing detail-specific functionality. This architecture comprises numerous devices at the cloud and fog levels, where scalability is essential and data become a key factor for achieving workflow objectives [56]. The alternatives for this challenge must consider centralized or decentralized orchestration, PLM-based or smart product, and the level of deployment of workflow manager (cloud, plant, workstation, etc.).

Chaining data from heterogeneous functions: Compared to the standard cloud model, where data are stored at a central point, in an IoT architecture, the cloud and fog devices operate together. Thus, the physical location and heterogeneity of information become an additional challenge [57]. The information and functionality are fragmented and where those data elements reside is essential for the correct WFM. Designers must chain this fragmented functionality into coherent workflows.

3.2. Requirements for Workflow Selection

During the stage of workflow selection, the most suitable workflows are selected for execution. A number of workflows might respond to a given set of needs associated with performing a certain process. In this phase, workflows are associated with a value depending on specific policies. The best feasible candidate is selected from an ordered set of workflows. The requirements at this stage are as follows:

Feasibility: This requirement questions not only whether a workflow is technically feasible but also whether it is possible to execute it considering the context in which it will be implemented. This implies verifying whether there are services with sufficient quality for each task in the workflow or if there are policies prohibiting the execution of a service in a workflow.

Valuation: Valuation consists of ordering workflows according to specific criteria (amount of tasks, computation cost, etc.). Valuation is usually performed according to the quality of the expected output.

Policy: A policy is a predefined set of rules that determines the validity of a workflow primarily based on cost or quality.

3.3. Requirements for Workflow Implementation

Workflow implementation is the stage where the allocation of resources and services is determined. Services are allocated to tasks in the managed domain of the workflow

according to service-level agreements and operational conditions as required. This phase precedes workflow execution. The following requirements become particularly relevant during workflow implementation.

Challenge 3. Collaboration: This requirement implies that candidate systems must be designed to generate and manage workflows in collaboration with other systems and their associated resources. The systems must standardize the output and communication between resources, meaning that the requirements for the partial workflows produced in terms of syntax and semantics must be met by each contender. There is also a need for the orchestration of workflows that involve multiple organizations. Once again, architectural constraints appear at this level. Heterogeneous edge cloud implementations affect the orchestration of resources from multiple administrative domains.

Openness: A system must be able to process workflow statements generated according to other partners' specifications and be tolerant of elements of expression that are outside the domain's and service's expertise. Distributed workflow orchestration is essential to exploit the potential of edge cloud architectures, where workflow elements manage different domains and coordinate in a hierarchical or peer-to-peer manner. In this environment, the concept of openness serves to assure that a service provider that encounters elements in a workflow that it cannot process does not: (1) fail with regard to the incomprehensible elements, (2) modify those elements, or (3) remove or discard those elements. Moreover, the complexity of heterogeneous application development and deployment processes needs to be hidden and a sophisticated level of abstraction has to be proposed. The solution must provide toolkits that simplify not only the tasks of resource/service discovery and monitoring but also the management of end-to-end workflows. In addition, usage policies and an adaptive migration mechanism need to be established [58,59].

Completeness: A system should provide a complete set of services for the generation and management of workflows. In case a system provides only a part of the required services during implementation, it must be extended through services from other providers that compensate for the lacking capabilities.

3.4. Requirements for Workflow Execution

In this phase, the workflow is executed or launched. The workflow execution requires an actionable and instantiated workflow. One of its functions is to translate the elements related to each assigned resource into service invocation data. Depending on the state of the resource, the workflow execution either transmits the invocation data or deploys the resource with the appropriate configuration. The main challenges identified here are as follows:

Challenge 4. Parallel execution capabilities: This challenge refers to the execution of multiple instances from the same workflow concurrently. It is also associated with multiple workflows requesting the same service at the same time, one workflow requesting multiple services at the same time, and so on. Parallelism can be approached from several points of view, depending on which actions can be carried out simultaneously in each scenario. Hence, it encompasses multiple instances of the same workflow running simultaneously, multiple workflows requesting the same service at the same time, or one workflow requesting multiple services at the same time.

Challenge 5. Asynchronous service execution: This challenge addresses a workflow's capability to accommodate asynchronous service communications. An asynchronous reply from a service task implies the provision of mechanisms to listen to the reply in order to continue the execution of a workflow. In comparison, a synchronous approach requires the task to stall until a reply is received from the service being invoked.

Challenge 6. Dynamic nature of microservice architectures: This challenge refers to a group of phenomena that occur due to the dynamic nature of microservice-based architectures. These phenomena include the following:

- **Dynamism:** Mixed cloud, edge, and IoT architectures are constantly changing. Edge conditions bring significant dynamism to service-based applications, in contrast to the relative stability of large data centers. Dynamic adaptation mechanisms, including run-time configuration, deployment, and switching, are essential for the smooth execution of workflows across the infrastructure [55]. According to [60], tolerance toward the loss of connectivity from the workflow orchestrator and the various controllers is typically not addressed by the technologies usually employed in WFM. Dynamism and the need to adapt workflows to changes can increase the likelihood of device failure and workflow orchestration issues [57]. The workflow orchestrator must enable near real-time scheduling of network resources such that it can adapt to changing service demands [61].
- **Rotation and unreliability:** This refers to the changes that can be experienced from the point of view of functionality and not so much from that of availability as in the previous case (dynamism). Similar to the dynamism challenge, edge resources are inherently volatile and are increasingly being used to support transient services. The functions offered by IoT or edge architectures are ephemeral, which imposes a much higher rate of change compared to cloud environments. This nature poses significant challenges in the different functions enabling workflow orchestration [55]. The description of resources and functionalities may not always be accurate, as they can quickly become obsolete, complicating reliable deployment and service guarantees.
- **Speed:** The microservices or serverless computing-based architectures that today's application systems work on are capable of delivering on-demand services with very fast response times. Hendrickson et al. [62] identified an important feature of serverless computing architectures: the ability to deploy new instances in milliseconds. These lightweight, rapidly deployable execution units are best suited for short-lived resources where failures are common. This presents a challenge for workflow orchestrators, who have to decide where to run a given function and reschedule services to cope with deployment speed and failures.
- **Discovery:** Microservice-based architectures and the constantly changing availability of services at the edge cause the discovery of available services at run-time to be an essential challenge for workflow orchestrators. The discovery of the available services and computing resources at the edge must go beyond predefined contracts and addresses. The probability of trying to contact a device that is no longer available is much higher at the edge, which causes device/service registration to be a pivotal aspect for assuring workflow execution efficiency [55].

Challenge 7. Security and privacy: IoT and edge networks are vulnerable to new types of attacks related to data flows. Attacks might come from inside or outside the company control domain and affect the management of workflows. The security requirements at the edge are numerous: control of data flows, authentication at different levels, authorization and access management for microservices, etc. Managing the security and privacy-related risks is a complex task for CPSs, although standard compliance guidelines are available for all industrial stakeholders to follow [63].

Mechanisms to correctly identify challenges related to WFM and their impact are necessary to support companies with the selection of the best WFM tools and languages to represent their manufacturing orders or recipes.

4. Strategies for Workflow Management in a Microservice Architecture

The challenges and requirements laid out in the previous section need to be overcome if workflow technology wants to reach further adoption in the industry. The use of workflows has the potential to be key in the management of complex industrial scenarios, in which systems work together to achieve the higher goals integrated into System of Systems (SoS). This is possible by establishing connections among systems, but, instead of predetermined

and fixed connections, workflows should support dynamic changes even at runtime, maintaining a clear overview of the SoS. Due to the current architectural and communication restrictions, such potential has not been explored in industrial environments.

Rather than performing an exhaustive comparison of all the strategies followed by WFMSs developed along the years (https://en.wikipedia.org/wiki/Workflow_management_system, accessed on 10 December 2022), this work focuses on comparing new and active solutions that follow the principles of microservice architectures. Each alternative has been developed by a different partner as a part of the research projects Productive 4.0 (<https://cordis.europa.eu/project/id/737459>, accessed on 10 December 2022) and Arrowhead Tools (<https://cordis.europa.eu/project/id/826452>, accessed on 10 December 2022), with the objective being the digitalization and automation of industrial solutions. The solutions are presented by describing their context, workflow representation, workflow language, and how the alternatives address the workflow-specific challenges.

The following sections describe the various microservice-based WFM approaches. Table 1 summarizes their characteristics according to how they address industrial requirements and challenges. Furthermore, Table 2 compares four different workflow management approaches. These investigative comparisons provide decision-making support with regard to the architecture, technologies, the language used to represent the workflow recipe, and other workflow-specific challenges.

Table 1. Workflow management challenges-and how they get addressed by each solution.

Workflow Challenges	Workflow Manager and Executor	Workflow Choreographer	WSO2 Enterprise Integrator	Node-RED Workflow Manager	FITMAN-CBPM
Workflow lifecycle stage support	Selection, implementation, and execution	Selection, implementation, and execution	Selection, implementation, and execution	Generation, selection, implementation, and execution	Generation, selection, implementation, and execution
(1) Workflow Modeling language	Finite state machines	BPMN and CPN	BPMN and BPEL	Workflow recipe in BPMN format	Workflow recipe in BPMN format. Services are semantically annotated using ontologies
(2) Heterogeneous infrastructure orchestrator scale	Centralized, orchestrated from a central point at the workstation level	Orchestrated from a central point	Centralized, orchestrated from a central point at the cloud level	Centralized, orchestrated from a central point at the workstation level	Centralized, orchestrated from a central point at the cloud level

Table 1. Cont.

Workflow challenges	Workflow Manager and Executor	Workflow Choreographer	WSO2 Enterprise Integrator	Node-RED Workflow Manager	FITMAN-CBPM
(3) Collaboration workflow tasks	Only able to assign tasks to software systems; no human operators unless they interface with a human-machine interface (HMI)	Only able to assign tasks to software systems; no human operators unless they interface with an HMI	Task types provided by BPEL and BPMN modeling languages are supported (service tasks, script tasks, user tasks, manual tasks, etc.)	Only service tasks are supported; no human operators unless they interface with an HMI	Ontological model for collaborative business process assessment. BPMN user tasks are supported intrinsically by Activiti.
(4) Parallel execution capabilities	Multiple instances of the same workflow and multiple service requests simultaneously	Multiple instances of the same workflow and multiple service requests simultaneously	Multiple instances of the same workflow are handled by the process engine and multiple services can run simultaneously using the parallel gateway node of BPMN.	Multiple instances of the same workflow and multiple service requests simultaneously	Multiple instances of the process can be launched and used by end users
(5) Asynchronous service request	Workflow Manager performs asynchronous calls for management services, while Workflow Executor depends on the workflow task's individual requirements	Service requests can arrive and be executed asynchronously and even in an order revealed on-the-fly if the recipe allows—as resource availability permits due to the Colored Petri Net approach	A service task can be mark as async at design time, an asynchronous service invocation follows the <i>Request/Acknowledge/Callback</i> pattern	The engine can detect BPMN tasks flagged as async, invoke the asynchronous service, detect dependencies, and executes accordingly	Asynchronous service-tasks are supported intrinsically by Activiti
(6) Dynamic nature of microservice architectures	The Arrowhead Framework supports the runtime discovery of available services and the re-orchestration of faulty endpoints	The Arrowhead Framework supports the runtime discovery of available services and the re-orchestration of faulty endpoints	The Arrowhead Framework supports the runtime discovery of available services and the re-orchestration of faulty endpoints	The Arrowhead Framework supports the runtime discovery of available services and the re-orchestration of faulty endpoints	Service discovery supported at design-time, provided by a central repository of services
(7) Computer security	Security provided by Arrowhead framework: AAA core system, authentication using X.509 standard certificates and authorization rules for granular access to each service	Security provided by Arrowhead framework: AAA core system, authentication using X.509 standard certificates and authorization rules for granular access to each service	Provided by WSO2 framework: users, roles, and certificates	Provided by Node-RED: Access to editor over https, authentication through username/password, and authentication against any OAuth/OpenID provider. Security between processes could rely on Arrowhead framework authorization core system	Provided by Activiti explorer: Access to editor over https in combination with basic authentication (user and password)

Table 1. Cont.

Workflow challenges	Workflow Manager and Executor	Workflow Choreographer	WSO2 Enterprise Integrator	Node-RED Workflow Manager	FITMAN-CBPM
Context	Workstation manufacturing workflows, mainly carried out using automated tools and machines on a factory shop floor	Any kind of automated manufacturing process that can be started and stopped by human operators. Furthermore, supporting automatic error handling in runtime	High-level enterprise-level processes with human intervention. Relies on other alternatives for workstation level	Any kind of dataflow or process except those involving human tasks (that require development). Specifically created for running on embedded systems	Collaborative business process management and SOA-based environment. The focus is the provision of tools for collaboration among the roles of BPM and manufacturing.

Table 2. Implementation features of the open source workflow technologies analysed.

Implementation Features	WSO2 Enterprise Integrator	Node-RED Workflow Manager	Workflow Manager and Executor	Workflow Choreographer	FITMAN-CBPM
Software license	Open source-Apache License 2.0	Open source-Apache License 2.0	Open source-Eclipse Public License 2.0	Open source-Eclipse Public License 2.0	Open source-GPL v3 Licence
Management interface	Design: Eclipse IDE Deployment: Web application Monitoring: Web application	Web Application/Tool GUI and REST API	REST API	REST API	Web application and REST API
Communication pattern	Processes interact by sending and receiving messages, connecting tasks in different pools and/or pools themselves	Handled at database level, the engine reads the BPMN structure to know the sequence flow	Request–response	Request–response	Follows the communication pattern provided by BPMN (sending and receiving messages)
Communication protocol	AMQP, MQTT, JMS, Amazon SQS, HTTP, and FTP	HTTP (other possible)	HTTP	HTTP	HTTP
Programming language	56% Java 38% JavaScript 6% other	JavaScript	Java 11	Java 11	Java, JavaScript, and Web Ontology Language
Runtime	WSO2 framework (JRE 8 behind)	Node-RED framework (Node.js behind)	Java Runtime Environment (JRE) 11	Spring framework	Activiti engine

4.1. Workflow Manager and Executor

The Workflow Manager and the Workflow Executor are two software systems designed to support workflows at several stages of their lifecycle, i.e., during selection, implementation, and execution. The system's main objective is to handle manufacturing workflows at a cell or workstation level, where several CPSs are deployed and provide their capabilities as services to other systems [23]. The systems are to be deployed in a manufacturing Industry 4.0 environment that follows a service-oriented paradigm. Specifically, the implementations

of the present system are designed to work with the Arrowhead framework mandatory core systems in a microservice architecture.

This solution relies on two separate software systems, each with their own goals and duties. Workflow Manager systems are designed to retrieve the product recipe information needed to start a particular workflow and command the Workflow Executor accordingly. In traditional factories, this information is stored and provided by a centralized system, such as an MES or factory control system. However, in the foreseen smart factories, this information would be decentralized and carried out by the products themselves, hence the use of the term smart products. A negotiation process will begin between the Workflow Manager and the smart product, which is the holder of the product recipe information, to match the available operations and costs associated with the manufacturing needs and goals of the stakeholders. Once an agreement is reached, the data will be processed and an order will be issued to the companion Workflow Executor.

Following the order, the Workflow Executor will activate the corresponding workflow, internally represented by a Finite State Machine (FSM). These workflows involve stages of manufacturing operations, such as assembly, drilling, or milling, that are to be performed by the workstation equipment through service calls. For each product to be manufactured in a workstation, where the Workflow systems are deployed, a dedicated FSM would be needed to automate the workflow. From a general point of view, the FSM's purpose is to specify in which order and with which parameters the workstation services should be requested. The entire process of interactions between the Workflow systems, smart product, and workstation equipment is summarized in Figure 4.

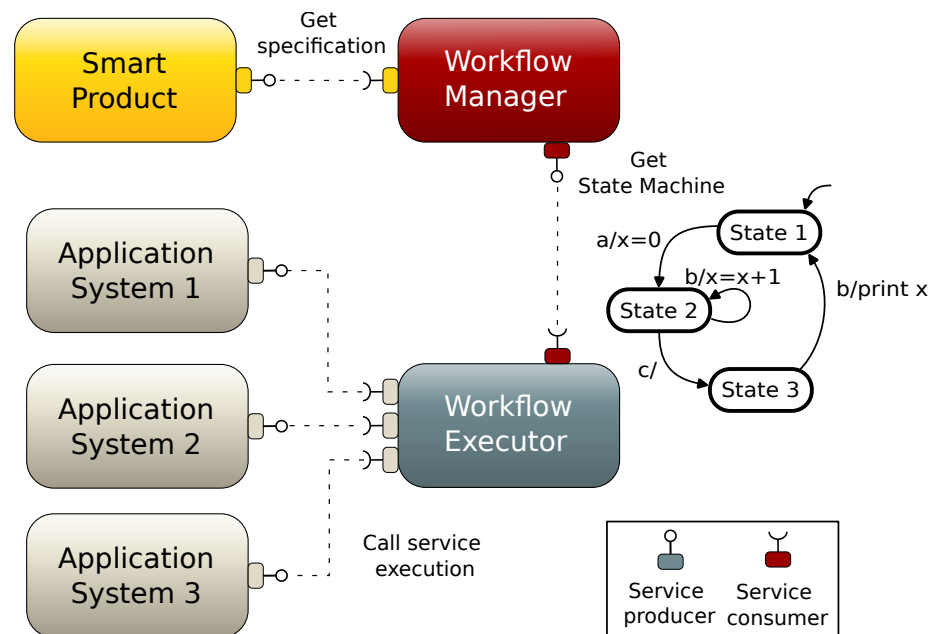


Figure 4. Workflow Manager and Executor service interactions.

The choice of the Workflow Manager and Executor solution implies several architectural decisions regarding the operation and management of workflows.

Starting with the web service composition strategy, these systems control the workflow centrally, are driven by the internal state machines, and consume the services according to their logic. In addition, the information relevant to each workflow step is routed through the Executor, which is in stark contrast with a choreography web composition strategy. The trigger signal for a workflow comes from an external system that consumes the Workflow Manager services, while working in a smart factory environment, such a signal is expected to arrive from a decentralized system, such as a smart product. In other cases, the command will arrive from a centralized system, such as an MES or ERP.

A few limitations of the capabilities of the process–task relation are imposed on the applications that use the Workflow Manager and Executor systems. The Workflow Executor is able to enact multiple workflows simultaneously. Each workflow executed is assigned an individual ID, which helps identify multiple workflows based upon the same underlying model, i.e., the same FSM. The ID is also communicated to the Workflow Manager to support the tracking of each workflow by third-party systems.

During runtime, while a workflow is being executed, it is possible to act upon it to stop, modify, resume, or discard it as needed. Both systems are designed with concurrency in mind and are able to handle multiple service requests and consume several services simultaneously. Since internal workflows are represented as FSMs, each workflow can only be in a single state at any given time. This feature provides determinism, i.e., the behavior of the system can be known in advance if the inputs and initial state are known, but it also has some shortcomings. The analysis of parallel FSMs is complex and other workflow representations can be more suitable for illustrating the state or modeling parallel tasks. Synchronous and asynchronous communications are possible using this solution, with the only limitations being imposed by the Java programming language chosen for the current implementation. The Workflow Manager, which handles communications with third-party systems, can harness the advantage of asynchronous communications at all times. Meanwhile, the Workflow Executor is contingent on the logic of each workflow, choosing between synchronous and asynchronous messaging according to each state action code.

Overall, the Workflow Manager and Executor provide a bare-bone and flexible solution to WFM. They provide workflow support without requiring major changes in infrastructure or being intrusive to other systems operating in the same environment. As they have been developed as research prototypes, the focus has been on functionality rather than reliability. Included are simple software constructions and a few dependencies to simplify future expansion, together with an open source license to promote community adoption. Nevertheless, they require one to be familiar with software programming to create the FSM and describe the state behavior. One of the main disadvantages of this solution occurs at the design phase of the workflow engineering lifecycle. The Workflow Manager and Executor systems do not explicitly support modeling state machines. However, once the state machine is modeled, it can be converted into software and added to the Workflow Executor through its web application programming interface (API).

4.2. Workflow Choreographer

The Workflow Choreographer is proposed to execute workflows based on predefined templates according to specific production processes. Based on the incoming production order, the Workflow Choreographer creates the so-called production recipe [64] and instantiates the production workflow. The production recipe uses predefined templates, which the operators must define according to the particular workflow. Based on the production steps, the Workflow Choreographer creates the workstation-specific Workflow Executors (see Section 4.1), which execute the associated workflows on the given workstation accordingly.

From an architectural perspective, the Workflow Choreographer sends requests to the Orchestration system, which is a core Arrowhead system [5], for the services that are required to execute the production recipe and subscribes to the related events provided by the Event Handler, which is an Arrowhead support system [65]. In addition, it communicates with the instantiated Workflow Executors. Based on the feedback service events from the Event Handler or the Workflow Executor, the Choreographer moves the workflow to the next appropriate step, according to the given recipe [66]. At the end of the workflow, it releases the reserved services, as shown in Figure 5.

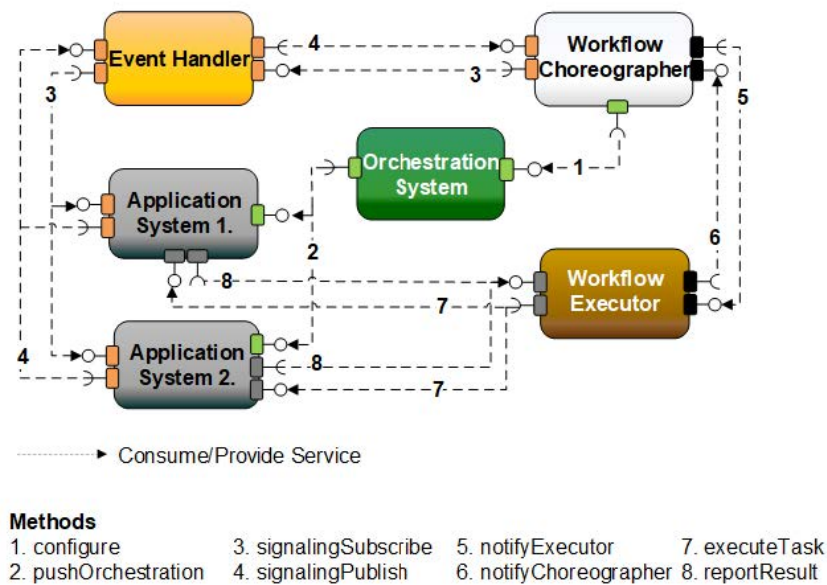


Figure 5. Workflow execution controlled through Workflow Choreographer service interactions [66].

The main novelty of the approach is that it also introduces the concept of multi-level workflow execution, in which the enterprise and production levels are separated [67]. A specific feature of the multi-level model is that the two levels use different modeling languages in parallel—BPMN [68] for the enterprise level and Colored Petri Net (CPN) [69] for the production level—thus the multi-level model retains their advantages and eliminates their disadvantages [67]. This separation was necessary to ensure the production workflows are transparent for operators while meeting the technical requirements of the identified core functions in line with Industry 4.0 expectations [67]. The two modeling languages are managed together at the program code level; this requires the creation and use of libraries that provide the syntax for the respective modeling languages that follow the programming language of Arrowhead.

Through the Arrowhead Management User Interface, workflows corresponding to BPMN objects can be graphically created using the production templates previously defined by operators. In these production templates, logic is implemented based on the CPN syntax [69]. To prove the concept, a reference implementation with the integration of several external solutions has been created [67], which is described in detail in Section 4.3. The reference implementation is written in Java, which is also the reference language for Arrowhead, and uses the Representational state transfer (REST) protocol for communication with external solutions. The reference implementation is also connected to the Arrowhead framework, which provides all the resources and functionality required for the Workflow Choreographer to work.

Since the workflow modeling has been carried out at the program code level, both synchronous and asynchronous operations can be achieved; furthermore, human-specific tasks can also be executed due to the BPMN syntax. Further, a system development methodology was also defined during the development of the Workflow Choreographer based on widely accepted standards and industry best practices [70]. This provides guidance for developing systems to be built into SOA-based SoS environments. Furthermore, based on the reference implementation, a prototype Workflow Choreographer was created and successfully tested under laboratory conditions [66].

An important difference from the concept presented earlier in Section 4.1 is that, in the case of the Workflow Choreographer, the Workflow Executors are always created for the task to be executed; therefore, the Workflow Executors are not pre-programmed and immutable supporting core systems. This causes distributed production and event response

to be much more efficient, as the workflow will be built for the given available resources taking into account the possible specific needs. Nevertheless, it also enables the possible inter-cloud workflow execution to be much easier as, due to heterogeneity, it may not be possible for a pre-programmed Workflow Executor to execute a task requested by a remote cloud. Previously created Workflow Executors can be kept in the case of frequently executable tasks. Consequently, they can only be used for specific tasks on the specified production line. This ability is advantageous, as a production line is typically designed to perform the same task.

The advantage of the Workflow Choreographer is that, although the modeling languages (BPMN and CPN) are currently defined, practically any other language can be merged at the code level in this way, which entails that the concept is flexible enough to support different solutions. Furthermore, the workflow models can be created graphically and can, therefore, be continuously monitored, which certainly causes the work of the operators to be easier. Perhaps a disadvantage is that the predefined templates must be created to enable the concept to work, which requires careful attention and somewhat complicates the process. However, at the same time, it is also essential because precision is a basic requirement in industrial environments. Consequently, another disadvantage is that the interoperability of the modeling languages at the program code level requires the implementation of the corresponding libraries, which are prerequisites for using the Workflow Choreographer.

4.3. WSO2 Enterprise Integrator

WSO2 is an open source enterprise platform for APIs, applications, and web services deployed locally and on the Internet [71]. The WSO2 Enterprise Integrator (WSO2 EI) is one of the products offered by the WSO2 platform. It enables enterprise services to collaborate dynamically between SOA-based systems [67]. Of its capabilities, we have selected the modules that are most closely related to the workflow lifecycle.

- **Tooling:** WSO2 Integration Studio (WSO2 IS) is among the modules offered by WSO2 EI. It is an eclipse-based drag-and-drop graphical development environment. It includes a visual tool palette, an option to import connectors to the tool palette, top-level integration constructs, and property views for complex configurations. Among its capabilities, there is the option to create BPMN or BPEL projects that enable the modeling of processes using the graphical interface provided by the tool and the generation of XML files.
- **Deployment:** WSO2 Business Process Server (WSO2 BPS) allows the user to deploy or undeploy business processes written in BPMN or BPEL through its web interface in a very intuitive manner. One has to simply select the file from the filesystem and upload it. Using the features, we can manage BPMN or BPEL instances (created beforehand) through the “Instances” dashboard and filter them according to their status (active, suspended, or completed). Some server settings can be customized in the “Configuration” dashboard, i.e., it allows one to add users, roles, keystores, and data sources. Finally, it also allows for the checking of the webserver statistics through the “Monitor” dashboard, which has some indicators such as memory usage, system uptime, average response time, and fault count.
- **Monitoring/interaction:** WSO2 BPMN Explorer enables interaction with deployed BPMN applications and allows the monitoring of the current status of workflows. Its main features are (1) a Dashboard tab where one can view statistics such as process instance count, the status of processes, and the status of task instances; (2) a Task tab to view, claim, and take action on BPMN tasks that require user intervention (e.g., filling a form); (3) a Processes tab to execute the deployed BPMN processes and view their diagrams; (4) a Monitoring tab to view the activity of specific process instances by providing the unique instance identifier, where one can see information such as start time, end time, and task state; and (5) a Reports tab to generate statistical reports of the current user’s actions in all the BPMN processes.

This alternative was implemented at the enterprise level in the multi-level hierarchical workflow execution approach presented in Section 4.2. The enterprise level, controlled by the WSO2 server, manages workflow orders written in BPMN and delegates on the Workflow Choreographer for workstation task execution at the production level, as presented in [67]. WSO2 orchestrates both human- and service-oriented tasks. Workflows can be designed and deployed easily using the tools and servers offered by the Enterprise Integrator package. This solution is able to manage multiple processes and instances simultaneously. Parallelism and concurrency are managed by WSO2, hiding the complex implementation of the concurrent task and relieving the user or the process developer of those duties. Process architecture and behavior are governed by the services that are to be orchestrated. WSO2 can implement most architectures and flow conditions, although asynchronous services are usually challenging. In addition to the design and deployment tools, the entire engineering lifecycle of the process is covered by WSO2, which enables the monitoring and control of the workflows during execution, as shown in Figure 6.

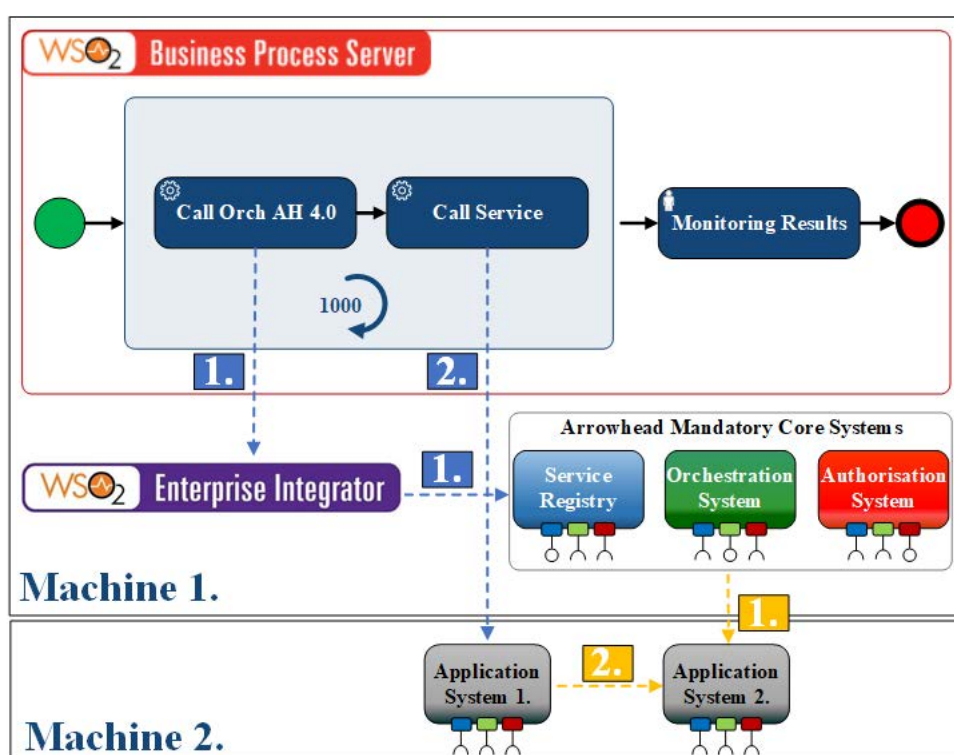


Figure 6. WSO2 Business Process Server and Enterprise Integrator managing workflow services [67].

We have identified certain advantages that differentiate WSO2 from similar tools: (1) Workflows can be designed intuitively through the graphical user interface using one of today's standard notation languages: BPMN or BPEL. (2) The tool is an Eclipse-based integrated development environment (IDE) that substantially reduces the learning curve, especially for Java developers. (3) It is a robust platform supported by its community on Slack, GitHub, and StackOverflow, among other channels. (4) It provides clear documentation and easy-to-run tutorials on the WSO2 website and on the IDE itself. (5) It has been implemented in production environments of many companies around the globe. In contrast, certain disadvantages have also been identified: (1) Apart from the existence of plenty of documentation and tutorials, which focus on "how to" instead of "why to", during the initial steps, it might be difficult to understand the right method to be used. (2) The open-source version still has some unresolved issues with the development tool that have only been addressed in the proprietary version. (3) The development tool can utilize a considerable amount of machine resources. (4) Customization can be complex.

As with the previous cases for many of the challenges present in today's industry, such as security and service management, the WSO2 alternative can rely directly on the Arrowhead framework to inherit its features [72]. Eclipse Arrowhead even enables secure autonomous management services [73].

4.4. Node-RED Workflow Manager

Node-RED is an open-source flow-based programming tool for integrating APIs, hardware devices, and web services into the IoT [74]. It provides a web browser-based flow editor; its core is built on Node.js and its flows are represented in the JSON format. Due to its open-source nature, we have implemented a Node-RED Workflow Manager tool (Node-RED WM) that is easily deployable in an embedded system [75]. The Node-RED WM mainly addresses the deployment and execution stages of the workflow engineering lifecycle.

Node-RED WM offers a set of rest endpoints for managing the uploading of recipes in the BPMN format, their instantiation, and their execution. First, the Node-RED WM uploads a BPMN file through its `/upload_recipe` endpoint. The BPMN recipe has to be created beforehand using other tools. Once the BPMN file is uploaded, the system stores it in a database and ensures it is available for execution through its `/start_process` endpoint. By invoking the endpoint and selecting the process recipe, an instance of the process will be executed. These endpoints are represented in Figure 7 together with the internal modules necessary to process each workflow task. A process can be launched multiple times and executed in multiple instances simultaneously. To do this, the Node-RED WM creates separated instances of the same process model and assigns a unique identifier to each instance. Several recipes can be uploaded simultaneously, enabling the orchestration of different processes at the same time. Workflows in execution are orchestrated by reading the BPMN file structure and requesting those services written in the BPMN process. The Node-RED WM reads the most relevant BPMN elements for process orchestration: the `startEvent` node to launch the process, the `sequenceFlow` elements to connect tasks, the `parallelGateway` to merge and split the process into branches, the `intermediateCatchEvent` elements to wait for events, the `serviceTask` to invoke service tasks, etc. In this way, Node-RED WM manages the process by controlling which task node is next in the flow until it reaches the BPMN `endEvent` node.

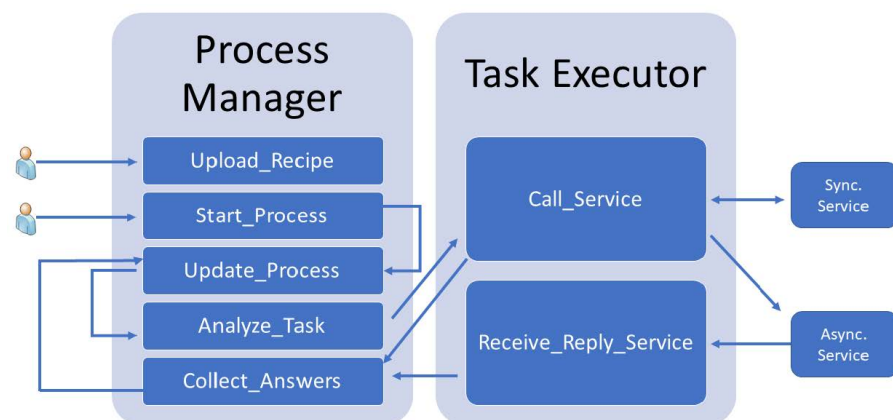


Figure 7. Node-RED Workflow Manager endpoints and modules [75].

This alternative enables the deployment of a workflow orchestrator in an embedded system such as an Arduino or Raspberry Pi. When compared to the WSO2 alternative, it consumes few resources, although its monitoring and control execution capabilities are currently limited. Node-RED WM can orchestrate only service-oriented tasks. Workflows in the BPMN format must be designed using BPMN modeling tools, but they are easily deployed and run using the REST endpoints created for that purpose. This solution enables

the management of multiple processes and instances simultaneously. Parallelism and concurrency are managed using IDs for each process instance and task. The status of each flow and its active task is managed using a database. The process architecture and behavior are conditioned by the service that is to be orchestrated. Node-RED WM can be employed to manage different architectures and flow conditions, although more BPMN elements need to be implemented for complete BPMN element coverage. Node-RED WM can also accommodate services with an asynchronous nature. Still, further improvements are necessary to cover the entire engineering lifecycle of the process. BPMN recipe modeling is performed using other tools and monitoring needs to be improved to enable a visual dashboard where one can track each process execution.

As presented above, the main advantage of this alternative is that it can be run in an embedded system. Although the technology readiness level of this workflow manager alternative is low, Node-RED offers the potential to enrich the solution by (1) accommodating other types of data formats or recipes; (2) integrating other frameworks and systems such as message brokers, databases, and protocols (OPC-UA); or (3) collaborating with other alternatives such as WSO2. As with the previous cases for many of the challenges present in today's industry, such as security and service management, the Node-RED WM can rely directly on the Arrowhead framework to inherit its features.

4.5. FITMAN-CBPM

FITMAN-CBPM is a web-based platform for the semantically enhanced design and execution of business processes in companies [43]. Targeted at the manufacturing domain and specially focused on collaboration among the roles of BPM and manufacturing [76], the platform is built using cloud-based architecture to allow one to access the user interface without needing to install anything on their machine. The database stores BPMN recipes, semantic/ontological data, and user data in a central server.

The workflow recipes are written in the BPMN format and are designed using COMPEL, a lightweight semantic workflow composition tool, which is based on Activiti. COMPEL uses ontologies to allow users to semantically annotate service tasks through the user interface [77]. Users select existing web services from a catalog and associate them with BPMN service tasks to automate their execution at runtime. Human tasks are supported intrinsically by the Activiti engine, allowing one to assign responsibilities to specific users or roles. The advantage of this approach is the support of standardized business processes using formal notation languages such as BPMN. Thus, the workflow recipes can be imported and/or exported from/to different platforms.

Collaboration tools are provided as a part of their platform within the Liferay portal, which is a web-based software that includes tools for document management, forums, and wikis. A REST API is also provided to manage the execution and monitoring of processes.

Among the limitations, there is the restricted service discovery feature that allows the selection of very specific service repositories due to technical reasons and the lack of support for REST services for binding them to service tasks, as the platform supports only the Web Services Description Language (WSDL).

5. Discussion

This section briefly analyzes the aforementioned alternative solutions for managing workflows. The different solutions follow different strategies to address the workflow challenges. These approaches are summarized in Table 1. On the other hand, the details of the technologies used to create each solution are listed in Table 2.

Starting with the comparison in Table 1 and considering the extent of the workflow lifecycle covered by each solution, we have identified a lack of support in the generation stage. The alternatives presented do not provide comprehensive toolsets for the engineering process *as is*; instead, engineers have a choice to model their workflows using third-party software before using a WFM solution. In general, modeling software tools provide the functionality to output a machine-readable document from a visual representation of

the workflow. Ideally, this should be understandable by any WFMS. Given the lack of standardization in the domain, this target is currently unachievable, as each software vendor has custom output formats. Therefore, the presented solutions require engineering effort to manually translate designs into the format used by each system unless they are integrated into a software suite that provides modeling.

A commonality emerges between the solutions proposed when it comes to the modeling the representation of workflows, as no solution supports more than a single modeling language. The solutions use different languages to represent and implement the workflow (such as BMPN, BPEL, CPN, etc.). We even found novel composed solutions in which each workflow has a different scope and uses a different language for representation. In such cases, the systems can collaborate such that each one handles the workflows they can interpret and executes the corresponding part in its domain. Even so, there are no solutions so far that can accept workflows in different modeling languages as their input. This can be attributed to the need for interoperability in the field, as there are too many language standards, which could have resulted from the absence of a language powerful enough to comply with the requirements of every use case. However, this can also be attributed to the differences in scope between the goals pursued by each solution.

The solutions use the same strategy for the architectural approach to web service composition. They all rely on an orchestration strategy to compose web services, reflecting a lack of the adoption of workflow choreography strategies. In each case, the controlling party of the orchestration is the software system provided as the WFM solution, which can interpret the workflow model and execute the calls for the services. Similarly, most solutions rely on a centralized system to provide the corresponding workflow recipe. These systems exist in organizations with hierarchical structures, where a system can globally coordinate operations but can present a single point of failure, which negatively affects reliability and availability. Having a decentralized source prevents these issues, but poses other problems when coordinating workflows and security.

Taking a closer look at the content of Table 1, we can see that all the solutions presented in this work have similar parallel execution capabilities. Being able to execute several workflows in parallel and perform multiple service requests is considered a must for a workflow support system, as this enables them to work with asynchronous services. Although no quantitative measures are provided, commercial systems are expected to provide a better performance with regard to the number of parallel workflow instances, service requests, and qualities of service (QoS). Moreover, a feature that seems less relevant and is lacking in most solutions is the *support for user or human tasks*. The investigated solutions generally do not focus on automating tasks that require user input and do not include users as resources. Instead, the focus has been on software systems that can work autonomously by design. Alternatively, if user input is required, it can be realized through the user interface of a machine on the shop floor, where the human resource is mapped to the services offered by a software system, thus abstracting the type of resource from the WFMS.

From a practical perspective, clear boundaries can be established between the solutions created from scratch, such as the Workflow Choreographer and the Workflow Manager and Executor, and those solutions based upon commercial platforms, such as WSO2 Enterprise Integrator and Node-RED Workflow Manager. The solutions without a platform at their foundations offered designers more flexibility, with fewer restrictions imposed by the tools and their predefined domains. They were designed from the ground up to serve production-level workflows. At the same time, the solutions forked from commercial platforms evolved from a general integration and coding tool to a workflow management application, which led to it being better suited for enterprise-level workflows.

In contrast, solutions based on enterprise-tested platforms have increased potential to attract companies. This is due to the extra support offered by professional organizations, with a developing team of engineers, the allocation of resources to marketing activities, and the trust provided by the organization's reputation. Other advantages over research

prototypes include a greater extent of documentation of the tools and an enthusiastic community of users. This last aspect should not be underestimated, as many open source software projects depend greatly on their community's contributions for success.

Considering the details of the actual solutions more deeply, Table 2 focuses on the implementation features of the workflow alternatives compared. These include the technical requirements (communication protocols, management interfaces, or communication patterns) and the software characteristics of the different alternatives (licenses, programming language, and runtime environment). Without delving deeper into the implementation details, it is important to note that all the solutions are provided through open source software licenses—even those supplied by private companies. This is a reflection of a prominent trend in the software industry, where new business models based on open source software are emerging, providing companies with a stable source of revenue.

Choosing among the microservice-based workflow management approaches needs a multi-criteria decision, with several alternatives satisfying a set of criteria. In order to better understand how each alternative responds to the criteria, we have employed the Analytic Hierarchy Process (AHP) method proposed in [78]. To validate the selection with the AHP method, we have used the XLSTAT extension of MS Excel.

The AHP procedure we implemented is the following. First, we built the problem hierarchy. In this step, we selected the alternatives against the criteria (as the main challenges) presented in Table 1. We included sub-criteria for two criteria elements. For "Parallel execution capabilities", we considered "Multiple instances of the same workflow" and "Multiple services simultaneously". For "Context", we considered "Workstation manufacturing workflows in a factory work shop", "Manufacturing process that can be managed by a human operator", and "Two level intervention (enterprise-Workstation)". In the next step, we—as four individual evaluators—analyzed the described hierarchy. As part of this process, we determined the *weight* assigned to each alternative concerning the criteria. We used a scale from 1 to 9. After that, we established *priorities* by assigning numbers to each node in the AHP hierarchy. A node is an element in the hierarchical diagram created with the AHP. These result in numbers between zero and one. After this step, we checked the consistency ratio of the judgments and adjusted the weights to fall within the recommendation provided by the AHP method (consistency ratio below 10%). For the different calculations during the process, we used XSLTAT [79]. At last, we obtained the final results of the AHP process. These results are summarized in Tables 3 and 4 and in Figure 8.

Table 3. Criteria Impact on decision (mean priorities by criterion %).

Workflow Challenges	Mean Impact %
(1) Workflow lifecycle stage support	14.14
(2) Heterogeneous infrastructure orchestrator scale	10.86
(3) Collaboration workflow tasks	8.59
(4) Parallel execution capabilities	14.79
(5) Asynchronous service request	11.80
(6) Dynamic nature of microservice architectures	12.37
(7) Computer Security	11.55
(8) Context	15.90

Table 3 shows that all the criteria have a similar impact on the final decision. No challenge seems to be more significant in the decision than the rest. Table 4 presents the mean priority per criteria and alternative. These percentages show the impact of each criterion when selecting a given solution-alternative. We can see that the total added percentages are similar for the first three alternatives, where the "Workflow Choreographer" collects the best score (27.15%). The alternative "FITMAN-CBPM" has the lowest percentages

(12.74%). Figure 8 is the graphical representation for Table 4. The figure shows that the alternative selection varies with the criterion considered. This implies that the selection of an alternative should be performed on a use-case basis. That is, depending on the challenge, each company considers more relevant for their production process.

Table 4. Alternative selection by criteria (mean priorities by alternative %).

Workflow Challenges	Workflow Manager and Executor	Workflow Choreographer	WSO2 Enterprise Integrator	Node-RED Workflow Manager	FITMAN-CBPM
(1) Workflow lifecycle stage support	1.96	4.98	2.73	2.35	2.12
(2) Heterogeneous infrastructure orchestrator scale	3.55	2.89	1.12	2.16	1.15
(3) Collaboration workflow tasks	1.50	1.50	2.32	1.29	1.98
(4) Parallel execution capabilities	3.35	4.24	2.59	2.66	1.95
(5) Asynchronous service request	2.81	2.94	2.16	2.03	1.85
(6) Dynamic nature of microservice architectures	3.40	3.40	2.40	2.40	0.76
(7) Computer security	3.26	3.26	2.18	1.94	0.90
(8) Context	3.11	3.93	4.55	2.28	2.03
Total Added %	22.95	27.15	20.06	17.11	12.74

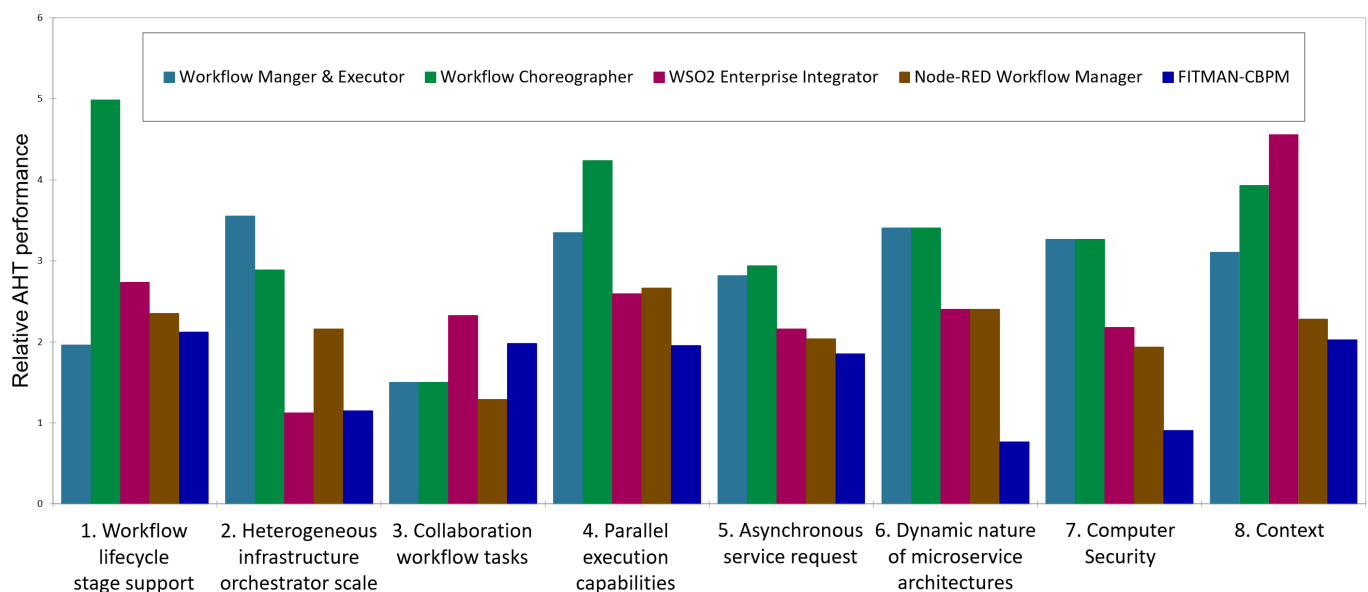


Figure 8. The results of the AHP alternative selection analysis selection for the microservice-based workflow management and execution alternatives

6. Conclusions

This paper analyzes a set of open source microservice-based WFM technologies to manage the dynamic market and Industry 4.0-driven requirements regarding production WFM and execution. The findings of this investigation clarify that these requirements can be met using open source microservice workflow technologies. To address the associated requirements effectively, a combination of two or more of the analyzed technologies have to be used.

The main contributions of the paper, besides identifying the industrial challenges regarding WFM and execution solutions are that it also suggests and compares various microservice-based solutions that are suitable for the task. The investigation of these microservice-based WFM and execution approaches reveals the actual ways in which

these solutions address the various industrial requirements and challenges. Furthermore, the comparison of the concrete implementations will help industrial stakeholders when choosing workflow solutions for various scenarios.

In conclusion, open source microservice workflow technology can be considered feasible for industrial usage, especially considering its flexibility, and support for operational and market dynamics. The implementation of these workflow management alternatives has been performed in lab-level use cases and not in the production lines of companies. Traditional production methods are still used in the industrial environment and microservice-based architectures have not yet been deployed. The main challenge now is to validate WFM technologies in real industrial contexts, where the flexibility and dynamism they enable with respect to personalized and automated production can be tested.

Author Contributions: Conceptualization, J.G.R., F.L., P.V. and J.D.; Methodology, J.G.R., F.L., P.V. and J.D.; Investigation, J.G.R., F.L., W.O., A.P., D.K., P.V. and J.D.; Software, J.G.R., F.L. and D.K.; Resources, F.L., P.V. and J.D.; Data curation, J.G.R., F.L. and P.V.; Writing—original draft preparation, J.G.R., F.L., P.V. and J.D.; Writing—review and editing, F.L., W.O., A.P., P.V. and J.D.; visualization, J.G.R.; Supervision, F.L., P.V. and J.D.; Funding acquisition, F.L., P.V. and J.D. All authors have read and agreed to the published version of the manuscript.

Funding: This research work has been funded by the European Commission, through the European H2020 research and innovation program, ECSEL Joint Undertaking, and National Funding Authorities from 18 involved countries under the research project Arrowhead Tools with Grant Agreement No. 826452.

Institutional Review Board Statement: No applicable.

Informed Consent Statement: No applicable.

Data Availability Statement: No 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.

References

1. Johnsson, C. ISA 95-how and where can it be applied. *Technol. Papers ISA* **2004**, *454*, 399–408.
2. Adolphs, P.; Bedenbender, H.; Dirzus, D.; Ehlich, M.; Epple, U.; Hankel, M.; Heidel, R.; Hoffmeister, M.; Huhle, H.; K²archer, B.; et al. Status report-reference architecture model industrie 4.0 (rami4. 0). In *VDI-Verein Deutscher Ingenieure eV and ZVEI-German Electrical and Electronic Manufacturers Association, Tech. Rep*; ZVEI e. V.: Frankfurt, Germany, 2015.
3. Lin, S.W.; Miller, B.; Durand, J.; Bleakley, G.; Chigani, A.; Martin, R.; Murphy, B.; Crawford, M. The Industrial Internet of Things Volume G1: Reference Architecture. Industrial Internet Consortium. 2019. Available online: <https://www.iiconsortium.org/pdf/IIRA-v1.9.pdf> (accessed on 8 June 2021).
4. Kozma, D.; Varga, P. Supporting digital supply chains by iot frameworks: Collaboration, control, combination. *Infocommun. J.* **2020**, *12*, 22–32. [[CrossRef](#)]
5. Delsing, J. *Iot Automation: Arrowhead Framework*; CRC Press: Boca Raton, FL, USA, 2017.
6. Ferreira da Silva, R.; Filgueira, R.; Pietri, I.; Jiang, M.; Sakellariou, R.; Deelman, E. A characterization of workflow management systems for extreme-scale applications. *Future Gener. Comput. Syst.* **2017**, *75*, 228–238. [[CrossRef](#)]
7. Ivančić, L.; Suša Vugec, D.; Bosilj Vukšić, V. Robotic Process Automation: Systematic Literature Review. In *Proceedings of the Business Process Management: Blockchain and Central and Eastern Europe Forum*; Di Ciccio, C.; Gabryelczyk, R.; García-Bañuelos, L.; Hernaus, T.; Hull, R.; Indihar Štemberger, M.; Kó, A.; Staples, M., Eds.; Springer International Publishing: Cham, Switzerland, 2019; pp. 280–295.
8. Carrera-Rivera, A.; Ochoa, W.; Larrinaga, F.; Lasa, G. How-to conduct a systematic literature review: A quick guide for computer science research. *MethodsX* **2022**, *9*, 101895. [[CrossRef](#)] [[PubMed](#)]
9. Kagermann, H.; Helbig, J.; Hellinger, A.; Wahlster, W. *Recommendations for Implementing the Strategic Initiative INDUSTRIE 4.0: Securing the Future of German Manufacturing Industry; Final Report of the Industrie 4.0 Working Group*; Forschungsunion Acatech: Frankfurt, Germany, 2013.
10. Russell, N.; Van Der Aalst, W.M.; Ter Hofstede, A.H. *Workflow Patterns: The Definitive Guide*; MIT Press: Cambridge, MA, USA, 2016.
11. Hollingsworth, D.; Hampshire, U. Workflow management coalition: The workflow reference model. *Doc. Number TC00-1003* **1995**, *19*, 224.

12. Hollingsworth, D. The workflow reference model: 10 years on. In *Proceedings of the Fujitsu Services, UK*; Technical Committee Chair of WfMC; Citeseer; Future Strategies Inc.: Lighthouse Point, FL, USA, 2004.
13. Wikipedia. Workflow Management Coalition—Wikipedia, The Free Encyclopedia. Available online: <http://en.wikipedia.org/w/index.php?title=Workflow%20Management%20Coalition&oldid=1114640089> (accessed on 23 January 2023).
14. van der Aalst, W.M. Don't go with the flow: Web services composition standards exposed. *IEEE Intell. Syst.* **2003**, *18*, 72–76.
15. Erl, T. *Service-Oriented Architecture: Concepts, Technology & Design*; Prentice Hall: Hoboken, NJ, USA, 2005.
16. Bashioum, C.; Behera, P.; Breining, K.; McCabe, F.; Brown, P.F.; Metz, R.; Hamilton, B.A. Reference Model for Service Oriented Architecture. OASIS Standard soa-rm, Organization for the Advancement of Structured Information (OASIS), 2006. Version 1.0. Available online: https://scholar.google.co.jp/scholar?hl=en&as_sdt=0%2C5&q=Reference+Model+for+Service+Oriented+Architecture&btnG= (accessed on 22 June 2021).
17. Cerny, T.; Donahoo, M.J.; Pechanec, J. Disambiguation and comparison of soa, microservices and self-contained systems. In *Proceedings of the Proceedings of the International Conference on Research in Adaptive and Convergent Systems, Krakow Poland, 20–23 September 2017*; pp. 228–235.
18. IBM Cloud Education. SOA (Service-Oriented Architecture). Available online: <https://www.ibm.com/cloud/learn/soa> (accessed on 13 August 2021).
19. Richardson, C. *Microservices Patterns: With Examples in Java*; Simon and Schuster: Shelter Island, NY, USA, 2018.
20. FIWARE Foundation, e.V. Available online: <https://www.fiware.org> (accessed on 25 August 2021).
21. Eclipse BaSyx. Available online: <https://projects.eclipse.org/projects/dt.basyx> (accessed on 25 August 2021).
22. Eclipse Arrowhead. Available online: <https://projects.eclipse.org/projects/iot.arrowhead> (accessed on 25 August 2021).
23. Garcia Represa, J.; Delsing, J. Autonomous production workstation operation, reconfiguration and synchronization. *Procedia Manuf.* **2019**, *39*, 226–234. [\[CrossRef\]](#)
24. Paniagua, C.; Delsing, J. Industrial Frameworks for Internet of Things: A Survey. *IEEE Syst. J.* **2020**, 1–11. [\[CrossRef\]](#)
25. Becker, J.; zur Muehlen, M.; Gille, M. Workflow application architectures: Classification and characteristics of workflow-based information systems. *Workflow Handb.* **2002**, *2002*, 39–50.
26. Leymann, F.; Roller, D.; Schmidt, M.T. Web services and business process management. *IBM Syst. J.* **2002**, *41*, 198–211. [\[CrossRef\]](#)
27. Meyer, S.; Ruppen, A.; Magerkurth, C. Internet of things-aware process modeling: Integrating IoT devices as business process resources. In *Proceedings of the International Conference on Advanced Information Systems Engineering, Valencia, Spain, 17–21 June 2013*; pp. 84–98.
28. Gartner, Inc. Market Guide for Intelligent Business Process Management Suites. Available online: <https://www.gartner.com/en/documents/3993207-market-guide-for-intelligent-business-process-management> (accessed on 16 June 2021).
29. Lyu, M.; Biennier, F.; Ghodous, P. Integration of ontologies to support Control as a Service in an Industry 4.0 context. *Serv. Oriented Comput. Appl.* **2021**, *15*, 127–140. [\[CrossRef\]](#)
30. Boschi, F.; Tavola, G.; Taisch, M.; Gepp, M.; Foehr, M.; Colombo, A. *PERFoRM: Industrial Context and Project Vision*; CRC Press: Boca Raton, FL, USA, 2019; pp. 1–31. [\[CrossRef\]](#)
31. Schnicke, F.; Kuhn, T.; Antonino, P.O. Enabling industry 4.0 service-oriented architecture through digital twins. In *Proceedings of the Software Architecture: 14th European Conference, ECSA 2020 Tracks and Workshops, L'Aquila, Italy, 14–18 September 2020*; Proceedings 14 2020; Volume 1269 CCIS, pp. 490–503.
32. Bellini, P.; Cenni, D.; Mitolo, N.; Nesi, P.; Pantaleo, G.; Soderi, M. High level control of chemical plant by industry 4.0 solutions. *J. Ind. Inf. Integr.* **2022**, *26*, 100276. [\[CrossRef\]](#)
33. Barz, M.; Poller, P.; Schneider, M.; Zillner, S.; Sonntag, D. Human-in-the-Loop Control Processes in Gas Turbine Maintenance. In *Proceedings of the International Conference on Industrial Applications of Holonic and Multi-Agent Systems, Lyon, France, 28–30 August 2017*; pp. 255–268. [\[CrossRef\]](#)
34. Suri, K.; Gaaloul, W.; Cucuru, A.; Gerard, S. Semantic framework for internet of things-aware business process development. In *Proceedings of the 2017 IEEE 26th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)*. IEEE, Poznan, Poland, 21–23 June 2017; pp. 214–219.
35. Steindl, G.; Kastner, W. Semantic Microservice Framework for Digital Twins. *Appl. Sci.* **2021**, *11*, 5633. [\[CrossRef\]](#)
36. Schaffer, E.; Schobert, M.; Reichenstein, T.; Selmaier, A.; Stiehl, V.; Herhoffer, M.; Mala, M.; Franke, J. *Reference Architecture and Agile Development Method for a Process-Driven Web Platform Based on the BPMN-Standard and Process Engines*; Elsevier: Amsterdam, The Netherlands, 2021; Volume 103, pp. 146–151.
37. Kayabay, K.; Gokalp, M.O.; Eren, P.E.; Kocyigit, A. *A Workflow and Cloud Based Service-Oriented Architecture for Distributed Manufacturing in Industry 4.0 Context*; IEEE: Paris, France, 2018; pp. 88–92.
38. Viriyasitavat, W.; Bi, Z.; Hoonsopon, D. Blockchain technologies for interoperation of business processes in smart supply chains. *J. Ind. Inf. Integr.* **2022**, *26*. [\[CrossRef\]](#)
39. Zubaydi, H.D.; Varga, P.; Molnár, S. Leveraging Blockchain Technology for Ensuring Security and Privacy Aspects in Internet of Things: A Systematic Literature Review. *Sensors* **2023**, *23*, 788. [\[CrossRef\]](#)
40. Leng, J.; Ruan, G.; Jiang, P.; Xu, K.; Liu, Q.; Zhou, X.; Liu, C. Blockchain-empowered sustainable manufacturing and product lifecycle management in industry 4.0: A survey. *Renew. Sustain. Energy Rev.* **2020**, *132*, 110112. [\[CrossRef\]](#)
41. Mazzola, L.; Waibel, P.; Kaphanke, P.; Klusch, M. Smart Process Optimization and Adaptive Execution with Semantic Services in Cloud Manufacturing. *Information* **2018**, *9*, 279. [\[CrossRef\]](#)

42. Borodulin, K.; Radchenko, G.; Shestakov, A.; Sokolinsky, L.; Tchernykh, A.; Prodan, R. Towards Digital Twins Cloud Platform: Microservices and Computational Workflows to Rule a Smart Factory. In Proceedings of the 10th International Conference on Utility and Cloud Computing (Ucc' 17). Assoc Comp Machinery; Ieee Comp Soc; Ieee Tcsc; Acm Sigarch; IEEE, Austin, TX, USA, 5–8 December 2017; pp. 209–210. [\[CrossRef\]](#)
43. European Commission, CORDIS. Future Internet Technologies for MANufacturing (FITMAN). Available online: <https://cordis.europa.eu/project/id/604674/> (accessed on 20 June 2021).
44. FIWARE. FITMAN-CBPM. Available online: <https://fimac.m-iti.org/dse1.php> (accessed on 15 June 2021).
45. De Panfilis, S.; Gusmeroli, S.; Rodriguez, J.; Benedicto, J. FIWARE for Industry: A Data-driven Reference Architecture. *Enterprise Interoperability: Smart Services and Business Impact of Enterprise Interoperability*; Wiley Online Library: Hoboken, NJ, USA, 2018; pp. 171–178.
46. FITMAN. Virtual Factory Experimentation Report. Available online: <https://cordis.europa.eu/docs/projects/cnect/4/604674/080/deliverables/001-D63FITMANVirtualFactoryExperimentationReportv10.pdf> (accessed on 20 June 2021).
47. In Proceedings of the 2017 IEEE International Conference on Software Architecture Workshops, ICSAW 2017, Side Track Proceedings, Gothenburg, Sweden, 5–7 April 2017.
48. Insfran, E. *Advances in Information Systems Development*; Springer: Berlin, Germany, 2022. Available online: <https://link.springer.com/content/pdf/10.1007/978-3-030-95354-6.pdf> (accessed on 28 January 2023).
49. van Veelen, J.B.; Holenderski, M. D2.2—State of the Art for Complex Workflow Generation. Available online: <https://productive40.eu/publications/> (accessed on 22 August 2021).
50. Lenhard, J.; Ferme, V.; Harrer, S.; Geiger, M.; Pautasso, C. Lessons learned from evaluating workflow management systems. In Proceedings of the International Conference on Service-Oriented Computing, Seville, Spain, 29 November–2 December 2017; pp. 215–227.
51. Albreshne, A.; Fuhrer, P.; Pasquier, J. Web services orchestration and composition. *Hewlett-Packard's Dev. Resour. Organ* **2009**, 46–52. Available online: <https://www.unifr.ch/inf/softeng/en/assets/public/files/research/publications/pdf/WP09-03.pdf> (accessed on 28 January 2023).
52. 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 Proceedings of the NOMS 2020-2020 IEEE/IFIP Network Operations and Management Symposium, Budapest, Hungary, 20–24 April 2020; pp. 1–6.
53. Ivanov, S.; Kalenkova, A. Comparing process models in the BPMN 2.0 XML format. In Proceedings of the Spring/Summer Young Researchers' Colloquium on Software Engineering, 2015; Volume 27, pp. 255–266. Available online: https://www.researchgate.net/publication/281371417_Comparing_process_models_in_the_BPMN_20_XML_format (accessed on 28 January 2023).
54. OMG. *Business Process Model Furthermore, Notation (BPMN)*; OMG: Milford, MA, USA, 2022. Available online: <https://www.omg.org/spec/BPMN/2.0/PDF> (accessed on 28 January 2023).
55. Vaquero, L.M.; Cuadrado, F.; Elkhatab, Y.; Bernal-Bernabe, J.; Srirama, S.N.; Zhani, M.F. Research challenges in nextgen service orchestration. *Future Gener. Comput. Syst.* **2019**, *90*, 20–38. [\[CrossRef\]](#)
56. Jiang, Y.; Huang, Z.; Tsang, D.H.K. Challenges and Solutions in Fog Computing Orchestration. *IEEE Netw.* **2018**, *32*, 122–129. [\[CrossRef\]](#)
57. Wen, Z.; Yang, R.; Garraghan, P.; Lin, T.; Xu, J.; Rovatsos, M. Fog Orchestration for Internet of Things Services. *IEEE Internet Comput.* **2017**, *21*, 16–24. [\[CrossRef\]](#)
58. Giang, N.K.; Blackstock, M.; Lea, R.; Leung, V.C. Developing IoT applications in the Fog: A Distributed Dataflow approach. In Proceedings of the 2015 5th International Conference on the Internet of Things (IOT), Seoul, Republic of Korea, 26–28 October 2015; pp. 155–162. [\[CrossRef\]](#)
59. de Brito, M.S.; Hoque, S.; Magedanz, T.; Steinke, R.; Willner, A.; Nehls, D.; Keils, O.; Schreiner, F. A service orchestration architecture for Fog-enabled infrastructures. In Proceedings of the 2017 Second International Conference on Fog and Mobile Edge Computing (FMED), Valencia, Spain, 8–11 May 2017; pp. 127–132. [\[CrossRef\]](#)
60. Rotsos, C.; King, D.; Farshad, A.; Bird, J.; Fawcett, L.; Georgalas, N.; Gunkel, M.; Shiomoto, K.; Wang, A.; Mauthe, A.; et al. Network service orchestration standardization: A technology survey. *Comput. Stand. Interfaces* **2017**, *54*, 203–215, SI: Standardization SDN & NFV. [\[CrossRef\]](#)
61. Velasco, L.; Castro, A.; King, D.; Gerstel, O.; Casellas, R.; Lopez, V. In-operation network planning. *IEEE Commun. Mag.* **2014**, *52*, 52–60. [\[CrossRef\]](#)
62. Hendrickson, S.; Sturdevant, S.; Harter, T.; Venkataramani, V.; Arpaci-Dusseu, A.C.; Arpaci-Dusseu, R.H. Serverless Computation with OpenLambda. In Proceedings of the Proceedings of the 8th USENIX Conference on Hot Topics in Cloud Computing, Denver, CO, USA, 20–21 June 2016; pp. 33–39.
63. Matta, G.; Chlup, S.; Shaaban, A.M.; Schmittner, C.; Pinzenöhler, A.; Szalai, E.; Tauber, M. Risk Management and Standard Compliance for Cyber-Physical Systems of Systems. *Infocommun. J.* **2021**, *13*, 32–39. [\[CrossRef\]](#)
64. Derhamy, H.; Andersson, M.; Eliasson, J.; Delsing, J. Workflow management for edge driven manufacturing systems. In Proceedings of the 2018 IEEE Industrial Cyber-Physical Systems (ICPS), Saint Petersburg, Russia, 15–18 May 2018; pp. 774–779.
65. Albano, M.; Ferreira, L.L.; Sousa, J. Extending publish/subscribe mechanisms to SOA applications. In Proceedings of the 2016 IEEE World Conference on Factory Communication Systems (WFCS), Aveiro, Portugal, 3–6 May 2016; pp. 1–4.

66. Kozma, D.; Varga, P.; Szabo, K. Achieving Flexible Digital Production with the Arrowhead Workflow Choreographer. In Proceedings of the IECON 2020 The 46th Annual Conference of the IEEE Industrial Electronics Society, Singapore, 18–21 October 2020. [CrossRef]
67. Kozma, D.; Varga, P.; Larrinaga, F. Dynamic Multilevel Workflow Management Concept for Industrial IoT Systems. *IEEE Trans. Autom. Sci. Eng.* **2020**, 1–13. [CrossRef]
68. White, S.A. *Introduction to BPMN*; IBM Cooperation: 2004. Available online: yoann.nogues.free.fr/IMG/pdf/07-04_WP_Intro_to_BPMN_-_White-2.pdf (accessed on 28 January 2023).
69. Jensen, K.; Rozenberg, G. *High-Level Petri Nets: Theory and Application*; Springer Science & Business Media: Berlin, Germany, 2012.
70. Kozma, D.; Varga, P.; Larrinaga, F. System of Systems Lifecycle Management—A New Concept Based on Process Engineering Methodologies. *Appl. Sci.* **2021**, *11*, 3386. [CrossRef]
71. WSO2, LLC. WSO2 Enterprise Integrator. Available online: <https://wso2.com/integration/> (accessed on 22 August 2021).
72. Plósz, S.; Hegedűs, C.; Varga, P. Advanced security considerations in the arrowhead framework. In Proceedings of the International Conference on Computer Safety, Reliability, and Security, Trondheim, Norway, 21–23 September 2016; pp. 234–245.
73. Maksuti, S.; Zsilak, M.; Tauber, M.; Delsing, J. Security and autonomic management in system of systems. *Infocommun. J.* **2021**, *13*, 66–75. [CrossRef]
74. IBM's Emerging Technology Services. Node-RED. Available online: <https://nodered.org/> (accessed on 22 August 2021).
75. Larrinaga, F.; Ochoa, W.; Perez, A.; Cuenca, J.; Legaristi, J.; Illaramendi, M. Node-RED Workflow Manager for Edge Service Orchestration. In Proceedings of the NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium, Budapest, Hungary, 25–29 April 2022; pp. 1–6.
76. Hachicha, M.; Fahad, M.; Moalla, N.; Ouzrout, Y. Performance assessment architecture for collaborative business processes in BPM-SOA-based environment. *Data Knowl. Eng.* **2016**, *105*, 73–89, Knowledge Engineering for Enterprise, Integration, Interoperability and Networking: Theory and Applications. [CrossRef]
77. Hachicha, M.; Moalla, N.; Fahad, M.; Ouzrout, Y. A maturity model to promote the performance of collaborative business processes. In Proceedings of the IFIP International Conference on Product Lifecycle Management. Springer, Doha, Qatar, 19–21 October 2015; pp. 112–124.
78. Saaty, T.L. What is the analytic hierarchy process? In *Mathematical Models for Decision Support*; Springer: Berlin, Germany, 1988; pp. 109–121.
79. Addinsoft. XLSTAT—Nalytic Hierarchy Process. Available online: <https://www.xlstat.com/en/solutions/features/analytic-hierarchy-process> (accessed on 23 January 2023).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.