IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 16 July 2024, accepted 11 August 2024, date of publication 21 August 2024, date of current version 2 September 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3446876

SURVEY

Dynamic Swarm Orchestration and Semantics in IoT Edge Devices: A Systematic Literature Review

BANANI ANURAJ[1,2,3], DAVIDE CALVARESI[1], JEAN-MARIE AERTS[2], (Member, IEEE), AND JEAN-PAUL CALBIMONTE[1,3]

[1]Institute of Informatics, University of Applied Sciences and Arts Western Switzerland (HES-SO), 3960 Sierre, Switzerland
[2]Department of Biosystems, Measure, Model & Manage Bioresponses Research Group, KU Leuven, 3001 Leuven, Belgium
[3]The Sense Innovation and Research Center, 1007 Lausanne, Switzerland

Corresponding author: Banani Anuraj (banani.anuraj@hevs.ch)

This work was supported in part by the Horizon Europe Project SmartEdge under Grant 101092908.

ABSTRACT In its dynamic evolution, the Internet of Things (IoT) has become increasingly pervasive in our daily lives, ranging from domestic appliances to industrial robots. This integration brings together people, processes, data, and devices, prompting new types of interactions among them. Besides acquiring data, these devices also have actuation and processing capabilities, making them susceptible of becoming autonomous entities with coordination potential. Given the inherent limitations of storage, power, or computation of IoT devices, delegation and cooperation strategies, including intermediary nodes in the network, can significantly optimize the usage of resources. Hence, this type of node can rely on swarm-inspired intelligence to orchestrate edge nodes, possibly with semantics-enabled behaviors. This study proposes a Systematic Literature Review (SLR) investigating different solutions and approaches for the orchestration of edge devices powered by declarative and semantic models of their affordances, goals, and capabilities. The SLR explores different aspects of the literature, including demographics, application domains, goals, requirements, scope, services, frameworks, and technologies, as well as challenges and future directions in the field. The purpose of this SLR is to provide software engineers, researchers, and innovators comprehensive insights into the present status of advancements in this area and a discussion of the unresolved issues and opportunities.

INDEX TERMS Internet of Things, swarm-based orchestration, dynamic orchestration, semantic sensor networks, edge devices, Web of Things.

I. INTRODUCTION

The pervasive impact of the Internet of Things (IoT) surge and the widespread adoption of smart devices exert tangible effects across nearly all industries, whether through direct or indirect means. IoT has transformed the traditional way of living into a high-tech-dependent lifestyle. Smart cities [1], [2], smart homes [3], energy efficiency [4], pollution control [5], intelligent transportation [6], and advanced industries [7] are all examples of such transformations brought about by the IoT. The driving forces of IoT include connectivity, data collection and analytics, cost reduction, efficiency, productivity gains, improved decision-making, innovation, new business models, and enhancements to the quality of life [8]. In the dynamic landscape of IoT and connected devices, achieving success demands service provision that stands out for reliability, scalability, and high performance, among other vital features [8].

Over the past decade, Cloud Computing and IoT have emerged as prominent technology solutions, garnering growing adoption and interest from both the scientific and industrial sectors. In fact, the integration and interplay of IoT and Cloud nodes enables an innovative paradigm delivering advanced services tailored for the aggregation, storage, and

The associate editor coordinating the review of this manuscript and approving it for publication was Chang Choi.

processing of IoT-generated data. While integrating IoT and Cloud introduces opportunities, it contends with specific constraints, such as bandwidth, latency, and connectivity.

The growing demand for facilitating communication between the cloud and IoT has fostered the birth of edge computing [9]. This approach involves placing computing and storage resources not just in the cloud but also closer to where the data originates. Edge computing enables intelligent resource allocation across IoT edge devices, and cloud devices, considering factors such as user experience, power consumption, computing load, performance, and cost. The hierarchical and cooperative edge-cloud architecture offers significant advantages by allowing the distribution of intelligence and computation, such as Artificial Intelligence (AI), Machine Learning (ML), and big data analytics; and in finding the best solutions while meeting specific constraints, such as managing tradeoffs between delay and energy consumption [10].

With the comprehensive availability/employment of IoT edge devices for sensing and actuation, enabling the orchestration and deployment of these systems becomes impelling. Given the complexity of the configuration of edge platforms, including the setup of device interconnections, integration of inputs/outputs, and the definition of tasks to be performed, fallback scenarios have to be planned [11].

Given the increasing autonomy of edge and IoT nodes, the idea of using self-organizing paradigms to govern their interactions has been studied in the literature. Swarm-based techniques for task orchestration—such as the use of ant colony optimization —have been explored to deliver outcomes with greater predictability [12]. A crucial factor in the success of swarm intelligence is its ability to maintain cohesive behaviors [13]. This approach is inspired by the natural swarming phenomenon, where groups of single entities, such as birds in a flock or ants in a colony, work together and follow simple rules to achieve complex and coordinated behaviors. By mimicking these natural systems, swarm intelligence leverages the collective actions of individual entities to solve complex problems, demonstrating how simple local interactions can lead to sophisticated global outcomes [14]. Swarm path planning using mobile edge computing has also been employed in use cases, including unmanned aerial vehicles (UAVs), in situations where conserving energy holds paramount significance [15].

Moreover, some literature also examines semantics-enabled behaviors. Semantics provide the ability to create abstractions that capture the essential capabilities, goals, roles, and tasks performed by edge and IoT nodes [16]. Semantic technology is actively employed to address the problem of interoperability [17]. This technology uses semantics to capture and represent the properties and complex relationships among IoT devices. By providing a common understanding of the data exchanged between different systems, semantic technology ensures that IoT devices can communicate and work together more effectively. It allows for the definition of data models and vocabularies that describe the functionalities and interactions of these devices, enabling seamless integration and coordination across diverse IoT platforms [18]. To the best of our knowledge, there are very limited studies that discuss on the integration of swarm intelligence and semantics. To name a few very recent studies, Sofia et al. have introduced semantics into their recent studies of decentralized container orchestration and identified swarm intelligence as one of their future studies [19]. Additionally, Wan et al. [20] proposed an architecture for context-aware production scheduling and control systems that uses ontology and reasoning technologies from knowledge engineering to improve system adaptability. Although semantic models have been discussed in this study, no swarm intelligence discussion is performed. Indeed, the combination of these edge nodes' semantics and dynamic swarm orchestration constitutes a novel research direction for the IoT domain within this context.

There are a few related reviews/surveys on this research direction, for instance, Mishra and Pandya [8] proposed a review of IoT applications, security challenges, attacks, and future visions. However, these authors do not address semantics or orchestration and analyze a few studies focusing on swarm behaviors with limited details. Moreover, in a recent Systematic Literature Review (SLR), Razian et al. [21] investigated and classified existing studies on service composition in dynamic environments under uncertainty. This SLR discusses only two studies with swarm intelligence and one study related to semantics. Sofia et al. share perspectives discussing the aspects that a dynamic orchestration approach should integrate to support an elastic orchestration of containerized applications [22]. To the best of our knowledge, no existing SLR addresses the targets or the research questions as a whole proposed in this study. More precisely, no existing SLR covers topics related to the integration of dynamic swarm orchestration in IoT edge devices and discusses to what extent are semantics used to represent and empower the interactions among IoT edge devices.

Nevertheless, even if only partially covering the topics proposed by this study, existing surveys have been discussed to confirm this review's findings. To mention a few, Kirimtat et al. [2] survey technologies covering a variety of research domains such as future trends and the current state of smart cities. Cao et al. [9] provide an overview of edge computing research. Abbas et al. [11] address technological developments in the area of mobile edge computing. Gagliardi et al. [23] discuss a review on virtual reality evacuation drills for safe built environments. Finally, Calvaresi et al. [24] elaborate on blockchain technology in the decision-making process of (multi-) agent-based systems. Section II mentions other SLRs [25], [26], [27], [28], [29], [30], [31] employing similar methodologies—yet addressing different research topics and applications.

To gain a more comprehensive understanding of the current panorama in this field, this paper presents an SLR elaborating on different aspects studied by the scientific community on the intersection between dynamic swarm orchestration of edge nodes using semantic descriptions of their capabilities and affordances. The objective of the SLR is to provide a comprehensive overview of this field, including state-of-the-art and open challenges to engineers, researchers, innovative managers, and practitioners. The contributions of this review are: (i) Conduct a SLR on dynamic swarm orchestration and semantics in edge nodes. (ii) Identify open challenges and propose directions for future research. (iii) Present key insights for future research based on SLR findings.

The rest of the paper is structured as follows: Section II provides an overview of the methodology used to perform this SLR. Section III introduces the planning phase of the review, encompassing the formulation of the protocol and articulation of the research questions. Section IV examines the results derived from the implemented methodology organized in alignment with the research questions. Section V engages in a discussion of the acquired results, including both those indicated in the primary studies and those inferred by the authors. Finally, Section VI summarises the key findings and insights, offering a conclusion to the paper.

II. SYSTEMATIC LITERATURE REVIEW METHODOLOGY

In this paper, we employ a methodology that prioritizes precision and the ability to reproduce results. The mentioned method is found in the procedural framework detailed by Kitchenham et al. [25]. The same methodology has been utilized in equivalent contexts [26], [27]. Figure 1 proposes a visual illustration of the adopted methodology, comprising three specific stages:

1) Review Planning: During this phase, the primary emphasis is on articulating the core fundamental questions and developing *Structured Research Questions* that constitute the foundation of the entire search protocol. This encompasses the alignment of criteria with the predetermined requirements of rigor and reproducibility, culminating in the validation of the protocol.
2) Conducting the Review: The review process involves the systematic implementation of predefined tasks, encompassing the assembly and meticulous selection of literature, detailed elaboration on the gathered literature, and the resolution of any discrepancies or conflicts that may emerge.
3) Dissemination: This comprises the detailed analysis, documentation, reporting, and summarising of the lessons learned, fostering a comprehensive understanding of the findings.

III. REVIEW PLANNING

This section formulates the SLR methodology to structure the research questions and craft the review protocol. The protocol includes a comprehensive discussion of the search strategy, inclusion and exclusion criteria, selection of study, biases, resolution of disagreements policy, and possible limitations of the SLR methodology.

A. RESEARCH QUESTIONS

As presented in Section I, the research community has proposed the usage of dynamic swarm orchestration and semantics in edge devices in recent years for different domains and stakeholders. Thus, the general research question can be interpreted within these terms as follows: *How is dynamic swarm orchestration performed, and to what extent are semantics used to represent and empower the interactions among IoT edge devices?* To delve more deeply into such an inquiry, we observed the Goal-Question-Metric (GQM) approach introduced by Galster et al. [28]. This methodology has been used in various other studies, including software measurement programs and VR simulators [23], [29], digital twins in manufacturing [30], tourism [31] and multi-agent systems and blockchain [24].

The dimensions explored in this study pertain to the domain of *intelligent* technologies and research. More specifically, we analyze existing contributions in these areas according to demographics and application domains, user classes, goals and requirements, scope granularity, frameworks/methodology, services and technologies, and semantic capabilities. Furthermore, we discuss their advantages, limitations, countermeasures, and future directions. Therefore, we formulated seventeen structured research questions (**SRQ**).

SRQ1. Demographics: *What is the temporal and geographical distribution of the research efforts?*
SRQ2. Domains: *In which application domains have dynamic swarm orchestration and edge semantics been employed?*
SRQ3. Users: *Who are the users that depend on dynamic swarm orchestration?*
SRQ4. Goals: *What are the objectives set for dynamic swarm orchestration?*
SRQ5. Requirements: *What are the requirements underlying the utilization of dynamic swarm orchestration?*
SRQ6. Granularity: *Which granularity characterizes the elaborated studies?* (i.e., *multi-swarm, swarm, node level)*
SRQ7. Characterizing nodes – *Which elements compose dynamically orchestrated swarms?*
SRQ8. Dynamicity: *To which extent are the orchestration and their nodes dynamic?*
SRQ9. Services: *Which functionalities are provided with dynamic swarm orchestration?*
SRQ10. Frameworks: *Which frameworks and implementations have been proposed to foster swarm orchestration?*
SRQ11. Technology: *Which technologies are integrated into the proposed solutions?*
SRQ12. Semantics enabled: *Which orchestration solutions employ semantics-based behaviours?*

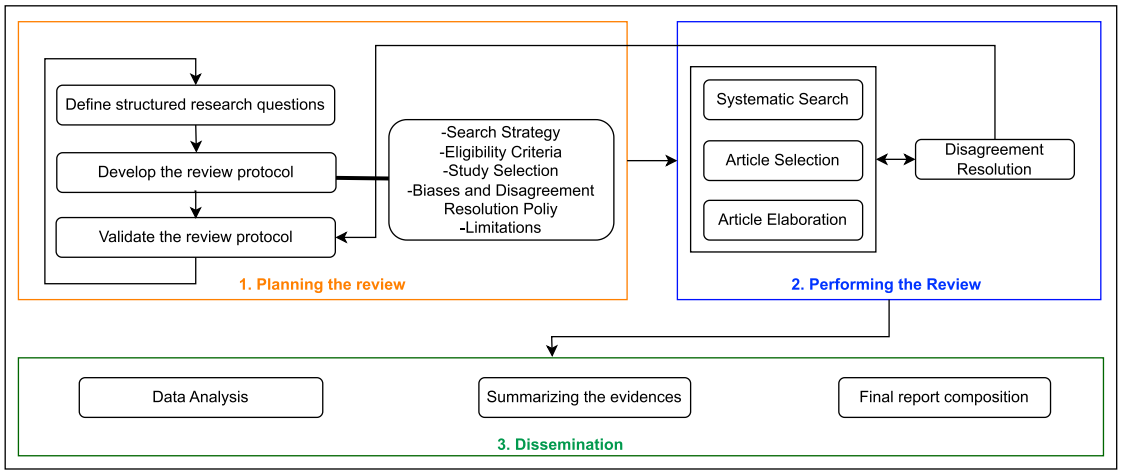

FIGURE 1. Systematic literature review methodology adapted from [25]; Highlighting the Planning, Performing and Dissemination of the review.

SRQ13. Semantics involvement: *What do the node(s) use semantics for?*
SRQ14. Advantages: *What are the strengths of employing swarm orchestration?*
SRQ15. Drawbacks: *What are the limitations of employing swarm orchestration?*
SRQ16. Countermeasures: *What are the proposed solutions for the drawbacks identified in SQR15?*
SRQ17. Future directions: *What are the future directions for swarm orchestration foreseen by the primary studies?*

B. REVIEW PROTOCOL

The review protocol consists of the **search strategy** (identify information sources/databases, search terms/queries), **eligibility criteria** (set inclusion and exclusion criteria for studies), **study selection** (describe the process for screening and selecting studies), **biases and disagreement resolution policy** (detail how discrepancies between reviewers will be resolved) and **limitations** (possible limitations of the SLR methodology). This protocol ensures the review is comprehensive, unbiased, and reproducible.

1) SEARCH STRATEGY

The *search approach* comprised choosing the following information sources: ScienceDirect,[1] IEEE Xplore,[2] CiteSeerX,[3] ACM Digital Library,[4] and Pubmed.[5] The choice of keywords was dependent on the reviewers' expertise and familiarity with the context of dynamic swarm orchestration and semantics in edge devices, and they include the following: dynamic swarm, orchestration, IoT, self-organizing, edge, semantics, ontology, and cyber-physical systems. Some keywords have been aggregated to increase the accuracy of the results. For example, dynamic swarm orchestration and semantics in IoT edge devices were expanded to different queries (see Table 1).

[1] http://www.sciencedirect.com/
[2] http://ieeexplore.ieee.org
[3] http://citeseerx.ist.psu.edu/index
[4] http://dl.acm.org/
[5] http://www.ncbi.nlm.nih.gov/pubmed

TABLE 1. SLR search queries and the number of results.

Queries	N. of results
dynamic swarm	ca. 80000
dynamic swarm (year 2000 onwards)	ca. 32000
dynamic swarm orchestration + IoT	ca. 6610
self-organising swarm orchestration	ca. 5730
self-organising swarm orchestration + IoT	ca. 5500
dynamic self-organising swarm orchestration	ca. 5000
dynamic orchestration swarm + edge	ca. 15700
dynamic orchestration swarm + agent	ca. 15600
dynamic orchestration swarm + multi-agent systems	ca. 5400
dynamic orchestration swarm + IoT + edge	ca. 17000
dynamic orchestration swarm + semantics + ontology	3820
"dynamic orchestration" swarm	163
"dynamic orchestration" swarm + semantics + ontology	500
dynamic orchestration + cyber-physical system + swarm + semantics	64

The list of papers to be considered was expanded by the inclusion of article sets derived from each combination of queries. Reviewers conducted a screening of the results for each query, evaluating the coherence of the articles with the research study. This evaluation specifically targeted the title and abstract, aligning with the criteria outlined in the following section.

2) ELIGIBILITY CRITERIA- INCLUSION AND EXCLUSION CRITERIA

The initial search gathered **64** papers. Next, further filtering was performed as shown in the following: (i) Prevent the inclusion of multiple papers, typically incremental, detailing identical work. (ii) Limit the time frame of the inquiry, for instance, by excluding less pertinent or excessively outdated works, taking into account technological advancements. (iii) Select works that contribute directly to the specific topic being investigated. (iv) Choose works that offer concrete theoretical or practical contributions, avoiding those that are solely visionary or speculative. The selection of papers adheres to similar criteria as [27] and [31].

Complying with the SLR guidelines [25], surveys have not been included in the primary studies in favor of the direct sources of the information. However, surveys are used in Section I (Introduction) and Section V (Discussion) to explain the technology and validate some of the review findings, respectively.

3) STUDY SELECTION

Three reviewers verified the mentioned inclusion and exclusion criteria, and each reviewer worked independently to select the relevant papers. Once individual filtering was complete, a final review was performed so that the refined list of papers was agreed upon by at least two reviewers. The refined list consisted of **47** papers, which would be henceforth referred to as *primary studies*.

4) BIASES AND DISAGREEMENT RESOLUTION POLICY

Reviewer biases and disagreement policies enable a thorough assessment of each task, aiming to mitigate biases and collaboratively resolve disagreements. In this context, the three reviewers cross-validated the inclusion/exclusion criteria throughout the selection process. Furthermore, regular meetings took place during the elaboration of the articles, facilitating the discussion and resolution of uncertainties.

5) POSSIBLE LIMITATIONS OF THE SLR METHODOLOGY

The identified risks/limitations of the adopted SLR methodology can be formalized as follows:

Accessibility – Possible selectable primary studies might not have been accessible or overlooked.

Timeliness – Very recent studies may not be tracked during the review writing time.

Clarity – Possible difficulty in extracting the necessary information due to the authors' lack of clarity in exposing the limitations of their studies. To tackle this issue, the authors used their knowledge to provide the readers with a better understanding and suggest extra details in the Discussion (Section V).

IV. REVIEW RESULTS AND ANALYSIS

In the following, we organize the outcomes of the SLR in alignment with the research questions outlined in Section III-A.

A. DEMOGRAPHICS

Referring to question **SRQ1**, Figures 2 and 3 show the temporal and geographical distribution of papers targeting dynamic swarm orchestration and semantics in IoT edge devices. Figure 2 illustrates the distribution of primary studies over the selected time window for this study. A subtle upward trend has been noticeable in recent years. This implies that the dynamic swarm orchestration and semantics research in IoT edge devices still appears to be a specialized or niche domain. Certainly, upon examination of Figure 3, the geographical distribution of the first authors' institutions (categorized by continent) corresponds to the dispersion of research groups in dynamic swarm orchestration. It is centered in Asia, North America (particularly the United States of America), and Europe, with a majority of work being in the latter.

B. APPLICATION DOMAINS

Regarding **SRQ2**, a visual representation in Figure 4 displays the various application domains chosen from the primary studies, demonstrating a significantly wide and expansive range of fields. For example, it ranges from cyber physical systems [20], [32], [33] to cognitive manufacturing [34] and security services [35]. Nevertheless, efforts have predominantly targeted IoT, UAVs, optimization, and healthcare across various domains. The IoT realm is composed of various sub-domains, including IoT edge clouds [36] and multi-cluster IOT Edge architecture [37]. UAVs also have a few sub-domains, such as IoT-Enabled drone swarms [38] and UAV-based wireless communication systems [39]. Optimization domains encompass dynamic multi-swarm particle swarm optimisation [40] and particle swarm optimization in dynamic environments [41]. Swarm-based applications include aerial swarms [42], swarm communication application [14] and swarm-breeding system application [43] Finally, healthcare use-cases are covered in [44], [45], and [46].

C. INTENDED USERS

Concerning **SRQ3**, Table 2 denotes how the selected primary studies identify and distribute intended user classes directly influenced by the specific application domains. The literature demonstrates that the majority of studies operate within the context of UAVs users, end users, and mobile users. Also, in the context of the type of studies, a significant number of studies are purely conceptual or general and do not address a specific use case (see Figure 5). In summary, the majority (40.43%) of the primary studies introduced various prototypes, 29.79% focused on technical or scientific concepts, and 29.79% of the selected papers incorporated extensively tested artifacts.

D. GOALS

Investigating **SRQ4**, we gathered and categorized the objectives outlined in the primary studies, as illustrated in Figure 6. Most of the papers addressed the theoretical foundations of

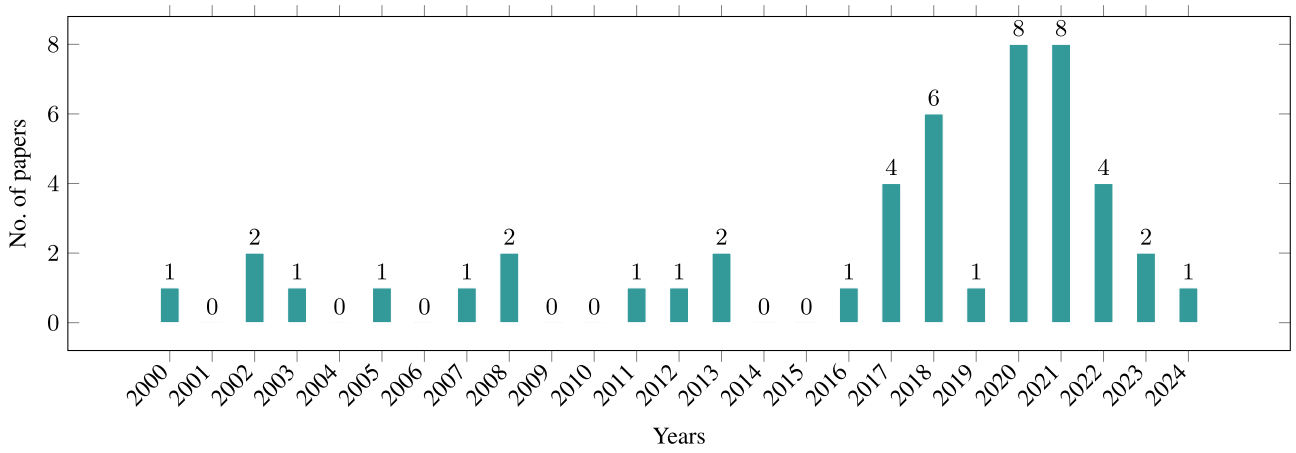

FIGURE 2. Total papers per year.

TABLE 2. Number of papers per type of users.

Type of Users	# papers	Studies
WoT users	1	[47]
Unspecified	21	-
UAV users	6	[13], [39], [42], [48], [49], [50]
Security users	1	[35]
Robot users	1	[51]
Physicians	1	[46]
Mobile users	3	[52], [53], [54]
IoT users	4	[19], [55], [56], [57]
Front-end users	1	[32]
End users	5	[33], [34], [37], [58], [59]
Emergency response team	1	[38]
Application users	2	[60]

swarm-based dynamic orchestration and semantics (*i.e.*, the primary emphasis of eighteen studies is on the conceptual aspects of the present state of the art or non-tangible systems). Within this category, we can mention [50], where an architecture in which a UAV swarm covers the energy demand of an IoT network is introduced. Another instance of theoretical foundations is addressed in [61], which offers a functional programming approach for robot swarms, incorporating well-defined properties to enable automated concurrency and distributed execution.

Practical implementations of dynamic orchestration have been proposed only by a few studies. For example, [58] delivers a multilevel workflow orchestration through a five-stage composition framework, catering to dynamic changes in user requests and ensuring scalability for automatic composition. The solution tackles the challenge of identifying and selecting the most appropriate service candidates through the semantic web service discovery (SWSD) mechanism. Moreover, the dynamical orchestration of Cloud resources is discussed in [62] and [63]. An improved model for efficient collaboration with the Linux operating system, prioritizing high-priority job execution is proposed in [62]. In [63] a customizable distributed scheduling approach tailored for individual applications is presented, to ensure scalability in setups with numerous nodes and intricate scheduling algorithms. An architecture for orchestration for the fog computing environment is presented in [64]. Reference [52] involves a discussion on selecting the suitable offload destination and orchestrating resources for multitasking.

In another classification, the goals were outlined to optimize dynamic environments. Liang and Suganthan [40] introduced a novel dynamic multi-swarm particle swarm optimizer (PSO). Morkevicius et al. [65] introduced a multi-objective optimization method for determining the optimal placement of services among the available fog nodes and PSO to search in both static and dynamic environments is discussed in [41].

The goals of the primary studies were also categorized under dynamic tracking (*e.g.*, for hazardous aerial plumes [38]). Additionally, the discussion centers on executing swarm tracking control for the flocking behavior of multi-agent autonomous helicopter models in [13].

Scheduling mechanisms were elaborated in [45] and [66]. Singh et al. [66] shared a newly developed scheduling approach for containers in big data applications, leveraging Docker Swarm and Microservice architecture. To offer an efficient resource scheduling scheme for critical healthcare tasks spanning an edge layer, fog node layer, and Cloud were discussed in [45]. Finally, Kim et al. [67] showcased a prototype implementation of IoT Cloud services specifically tailored for efficient cooling management in smaller server room environments.

E. REQUIREMENTS

Concerning **SRQ5**, we extracted the requirements articulated in the primary studies. The development of the primary

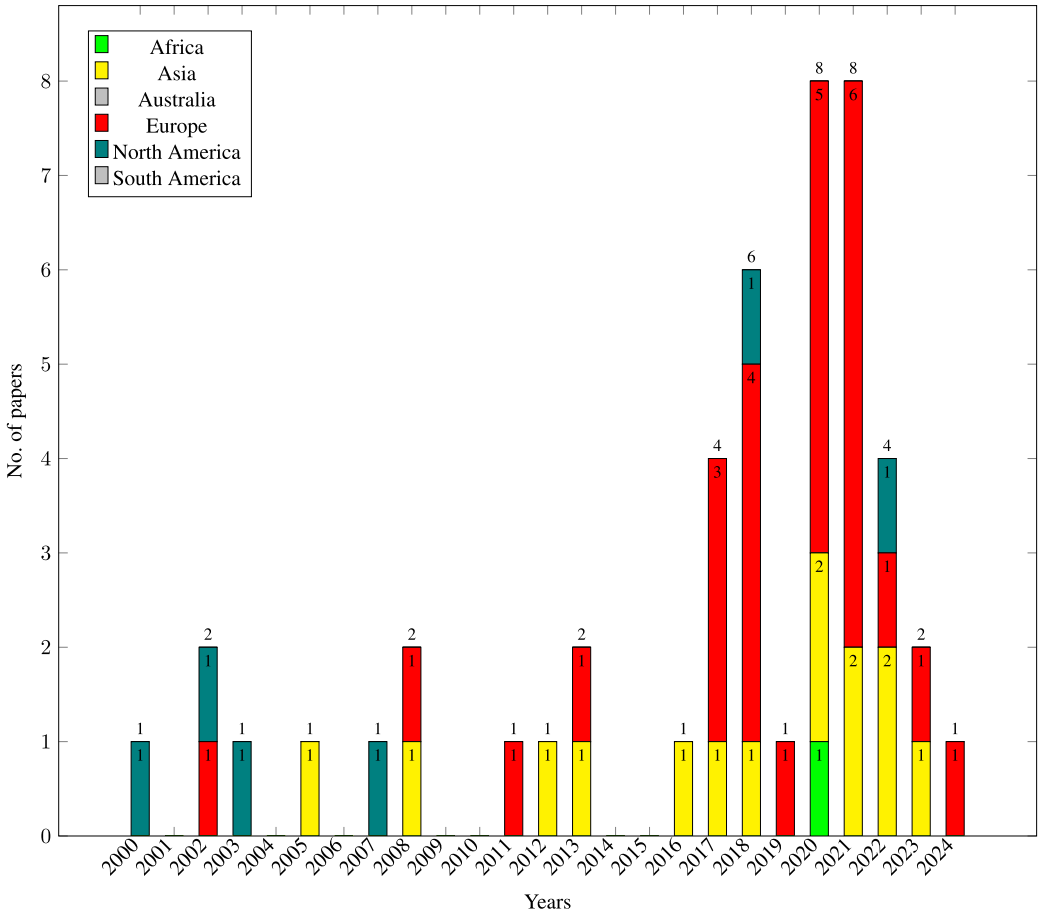

FIGURE 3. Number of papers per continent per year.

features encapsulated by these requirements is classified into the following categories:

- **Functional Requirements** are used to describe requirements that have a direct impact on the behavior or functionality of the platforms (see Table 3).
- **Architectural Requirements** refer to specifications that influence the system or the back-end components of the platforms (see Table 4).
- **Front-end Requirements** are outlined as specifications that exclusively concern the user interface or visual elements of the platforms (see Table 5).

Figure 7 visualizes the distribution of different requirement types elicited from the primary studies. The key concentration of the authors in the elaborated papers is primarily directed at functional (51.06%) and architectural (42.55%) requirements. Only 6.38% of the studies explicitly formalized requirements related to the front end.

A significant amount of functional requirements revolve around dynamic scalability [14], [60], dynamic orchestration [62], [63], and resource discovery [44] which are critical for orchestrating of IoT edge devices [57]. Indeed, these requirements correlate with the majority of goals in the primary studies (See Section IV-D) to enable dynamic orchestration. Dynamic orchestration is achieved by Yeh and Yu [62] using the Linux operating system to expedite the execution of high-priority jobs to a significant degree. Also, a mechanism for automatically discovering services and resources to efficiently deploy nano services on local IoT nodes in real-time is discussed by Islam et al. [44], which correlates with the majority of the application domains (IoT). Moreover, architectural requirements of implementing a smart energy IoT-Cloud services [67] and front-end requirements of establishing a financially stable charging station offering energy coverage to the IoT [50] also show a correlation to the majority of the application domain, *i.e.*, IoT (See Section IV-B).

F. SCOPE GRANULARITY AND NODE CHARACTERISTICS

This section covers the scope granularity of a swarm-based approach (**SRQ6**) and the characteristics of the nodes in the swarm (**SRQ7**).

Studying **SRQ6**, we have classified the primary studies according to their scope granularity to identify whether a swarm-based approach was used and if it was multi-swarm. The advantage of a multi-swarm approach compared to no swarm or a single swarm is the ability to exchange

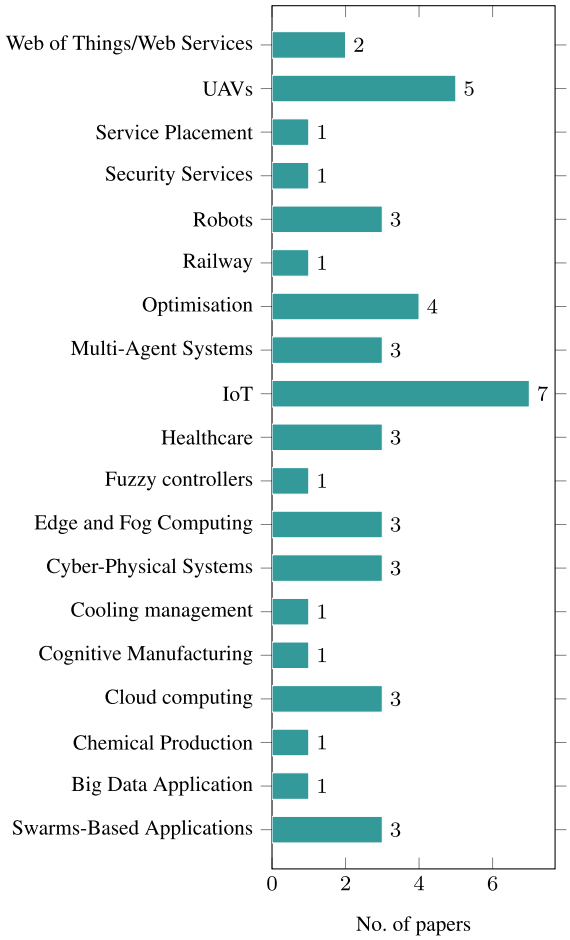

FIGURE 4. Contributions per application domain.

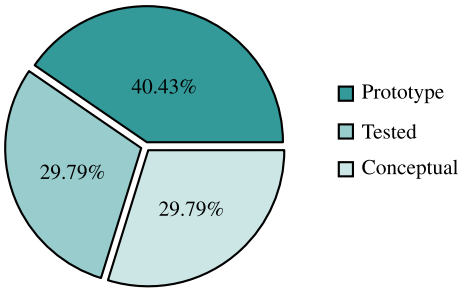

FIGURE 5. Types of studies according to level of abstraction.

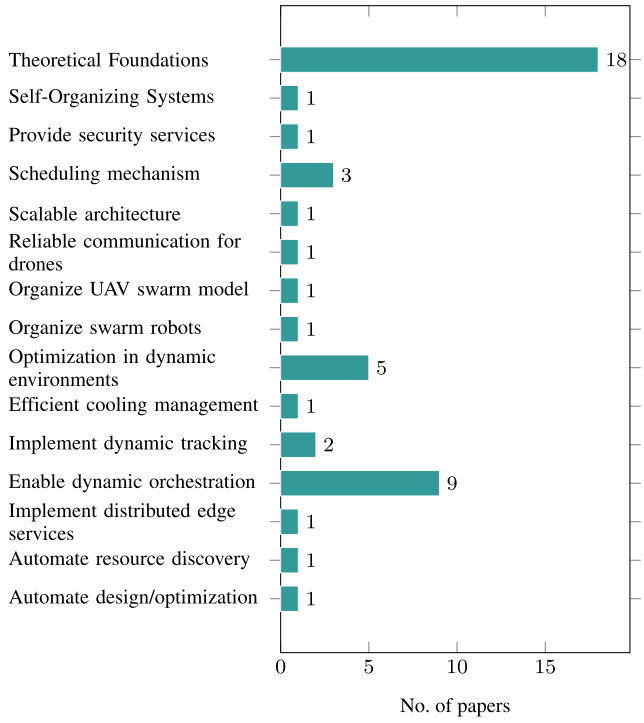

FIGURE 6. Goals of the primary studies.

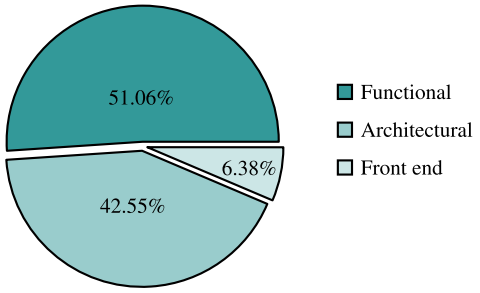

FIGURE 7. Types of requirements.

information among the swarms [40] and multi-swarm optimization is tailored for dynamic optimization problems rather than looking for local optimum [75], [76]. Figure 8 shows a graphical representation of the results. The primary focus of the authors in the detailed studies is predominantly centered on swarm-based (46.81%), not swarm-based (46.81%), and very few studies (6.38%) discussed multi-swarm approaches.

It is also directly related to **SRQ7** in which nodes are categorized to identify which elements compose dynamically orchestrated swarms in swarm-based and multi-swarm-based approaches. This is illustrated in Figure 9. [37], [46], [48] use edge nodes to compose orchestration in swarm-based approach. Fog nodes are used in [54], [57], and [65].

The manager and worker node concept is used in [66] and [67]. The manager node oversees all membership and allocation processes, while worker nodes execute swarm-based services in Docker Swarm. The swarm controller that generates a swarm center and the tracking controller that tracks the swarm center are implemented in [13]. Master and slave drones are used in [38]. The drone swarm is organized in a grid pattern with drones situated in four quadrants, and one drone functions as the master drone responsible for coordinating communication with the ground-based command and control system. Slave drones, located in the left, right, and rear quadrants, transmit their data to the master drone via WiFi, which then routes this information to a central command and control computer. Consequently, the entire swarm moves left in the direction of the smoke, with each drone identifying and surrounding the source of the smoke.

TABLE 3. Functional requirements.

Study	Functional Requirements
[60]	Enable automated scalability and orchestration.
[48]	Implement self-learning and self-decision-making abilities.
[49]	Provide resilience and Quality-of-Service.
[44]	Provide capabilities of virtualisation and resource discovery.
[66]	Implement precise load-balancing mechanisms.
[68]	Provide automated design and optimization of self-organizing fuzzy logic controller.
[45]	Provide scheduling, balancing, and prioritization.
[63]	Perform dynamic orchestration.
[14]	Provide dynamic scalability in a swarm system.
[46]	Implement orchestration, adaptation, and health monitoring.
[53]	Perform a logical operation by combining components dedicated to motion tracking analysis, predicting movements, and visualizing the anticipated outcomes.
[62]	Perform dynamic resource orchestration.
[56]	Enable real-time-compliant negotiation in multi-agent systems.
[69]	Provide robot swarm interaction and web services ecosystem.
[70]	Use semantic web technologies to facilitate the transformation, analysis, and reasoning capabilities over the dataset.
[40]	Introduce dynamic multi-swarm particle swarm optimiser.
[35]	Use fog nodes with lightweight virtualization technologies to establish the infrastructure for service provisioning.
[54]	Establish data locality, response latency, reliability tolerance, and minimum security satisfaction levels. Translate the logical workflow design into a physically executable workflow across resources of Fog appliances.
[71]	Use multi-agent systems, service-oriented systems, holonic systems, and self-organization.
[72]	Utilise a dynamic neighborhood strategy, dynamically update particle memory, and apply one-dimensional optimization to address multiple objectives.
[61]	Use functional robot swarms.
[57]	Provide heterogeneity, scalability, and adaptability.
[73]	Perform dynamic deployment, orchestration, and monitoring solutions for distributed applications. Enable automatic installation of new behaviors based on environmental triggers.
[19]	Outline the optimal infrastructure to support the operation of next-generation Internet applications.

TABLE 4. Architectural requirements.

Study	Architectural Requirements
[51]	Perform the formation of swarm robots and self-organization.
[55]	Adopt established open standards for the architecture design and demonstrate its feasibility by implementing a smart telematics application deployed within a vehicle.
[13]	Provide swarm tracking control for flocking of multi-agent autonomous helicopter models.
[74]	Implement Gaussian local search and differential mutation for dynamic optimization.
[52]	Use particle swarm algorithm to obtain solutions that approximate the optimal outcome.
[47]	Formulate the Web Things (WT) allocation as a multi-objective optimization problem and propose a graph-based heuristic.
[37]	Utilise a multi-cluster edge layer configuration that incorporates separate, independent edge node clusters at the local level.
[59]	Set up the ACCORDION infrastructure to fulfill the NextGen mobile online gaming requirements, reducing latency between servers and clients for improved user experience.
[42]	Use self-organising TDMA (STDMA) protocol for aerial swarms.
[32]	Propose using software containers and cloud-agnostic orchestration facilities to empower system operators.
[36]	Implement a framework that enables trustworthy orchestration for edge Clouds.
[34]	Established a mapping of the physical and virtual holon, embodying it as an intelligent agent present at the edge, fog, and cloud levels.
[64]	Provide a virtual environment enabling fog nodes to run virtualized and containerized applications and services.
[41]	Integrate optimization to allow PSO to search in both static and dynamic environments.
[67]	Implement a smart energy IoT-Cloud service using the open-source miniaturized playground for IoT-Cloud services. Leverage container-based service orchestration with Docker Swarm.
[65]	Employ the integer multi-objective particle swarm optimization method to identify a Pareto set of non-dominated potential service distributions.
[39]	Adopt a model of virtual forces to facilitate control over the movement of each UAV by manipulating a set of persistent and mutable parameters within the system.
[43]	Apply dynamic swarm behavior patterns using an interactive evolutionary algorithm.
[33]	Facilitate the expression of application logic for a self-organizing cyber-physical system independent of deployment specifics.
[20]	Develop context-aware scheduling and control architecture.

The nodes are also characterized as worker and balancer nodes [14]. To execute a work in the system, this work should be requested by creating `WorkerRequests` swarms. `WorkerRequests` consist of `doChooseWorker` (executed in Balancer nodes), `executeWork` (executed in Worker nodes) and `taskDone` (executed in Balancer after a task is completed).

G. DYNAMICITY

In **SRQ8**, we identified to which extent the orchestration is dynamic in the primary studies. All the primary studies discuss dynamism in one way or another. For instance, [58] addresses the dynamic changes of user goals, and dynamic route planning involving speed and charging guidance is covered in [48]. Additionally, Seiber et al. [38] explore the coordination of drones as a swarm, dynamically adapting their positions to track the outer perimeter of the plume.

H. SERVICES

In **SRQ9**, we address the services the primary studies provide. Figure 10 provides a visual representation of the services.

TABLE 5. Front-end requirements.

Study	Front-end requirements
[58]	Develop an improved five-stage composition framework designed to facilitate the dynamic modification of user requests and enhance scalability for automatic composition.
[38]	Use augmented sensor-based drones.
[50]	Establish a financially stable charging station (CS)-UAV association that ensures profit for both parties, offering energy coverage to the IoT.

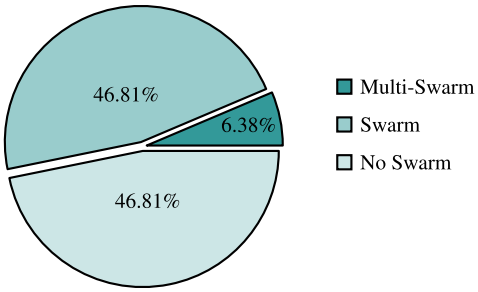

FIGURE 8. Scope granularity.

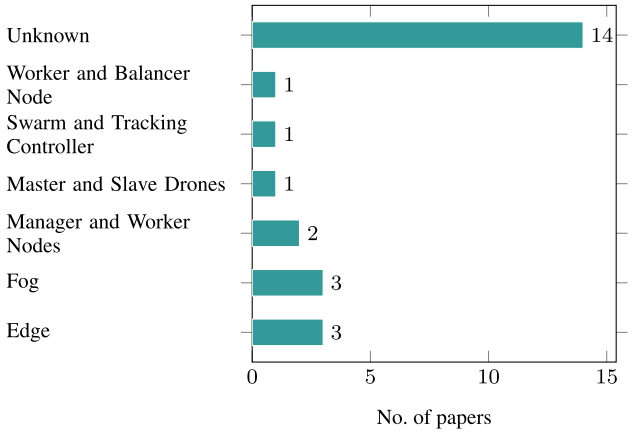

FIGURE 9. Node categorisation.

Microservices are used in [53]. Microservice architecture is a relatively newly formed concept in the area of software models that involves splitting a monolithic application into separate processes that implement a single function. Microservices are also used in [19], [32], [33], [44], [54], [55], [57], and [66].

Services related to the Cloud such as resource monitoring, automated testing and resource migration are covered in [36] and [46] and supporting the seamless and reliable operation of IoT-Cloud services is discussed [67]. Container-based orchestration is implemented to enhance the automated and reliable operation of IoT-Cloud services.

Solutions based on Web services are covered in [58], [69], and [71]. The functionality of robots is exposed as Web services in [69], and different applications may invoke such services whenever needed. Likewise, robots can also access external information or inferred knowledge either

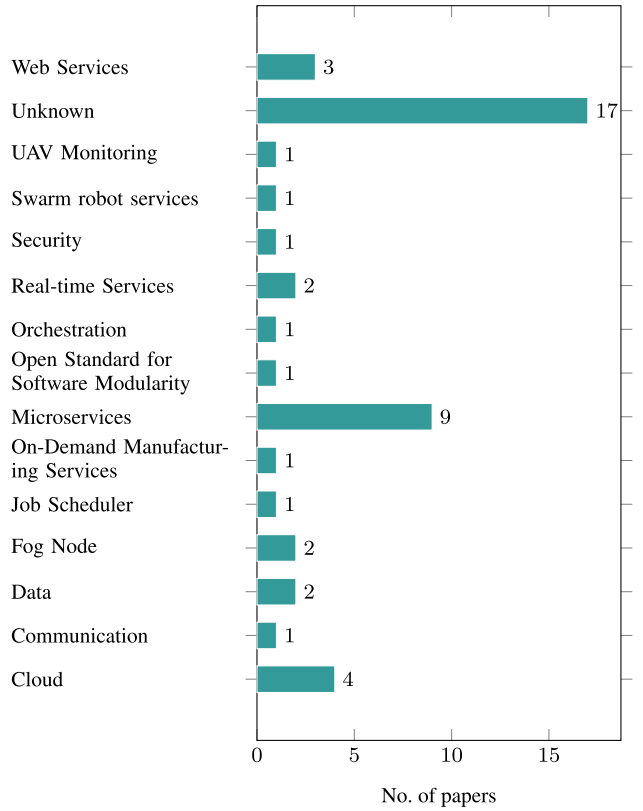

FIGURE 10. Type of services in primary studies.

from robots in the environment or by discovering external services.

A few other service categories include security services [35], on-demand manufacturing services [34], and data avenue services [60]. Security services such as Intrusion Detection and Prevention Systems (IDPSs) are implemented within the fog infrastructure, emulated using a Raspberry Pi-4 device in [35]. The Cloud manufacturing (CMfg) model facilitates easy and on-demand network access to a shared set of configurable manufacturing resources, including software tools, equipment, and capabilities. These resources can be rapidly launched with minimal management effort or service provider interaction, as outlined in [34]. The data avenue services discussed in [60] facilitate the transfer of files or directories of files between different data storage systems with various storage access protocols.

I. FRAMEWORKS

SRQ10 covers the frameworks implemented in the primary studies. Table 6 summarizes the overview of the frameworks in the studies.

For each contribution, we provide a short description of the framework proposed and/or presented in the paper at a high level. Given the diversity of the different work surveyed, these frameworks span different aspects, ranging from UAV configuration approaches to IoT coordination approaches.

TABLE 6. A high-level overview of existing frameworks.

Study	Frameworks
[60]	The Microservices-based Cloud Application Level Dynamic Orchestrator (MiCADO) supports multiple heterogeneous federated clouds. MiCADO includes both an optimized deployment process and run-time orchestration.
[58]	SWSD to select the most suitable service candidates.
[48]	An agent-based architecture is implemented to monitor and control swarms of resource-constrained UAVs. Most tasks involving extensive processing are offloaded to the Smart Cities (SC) Leader UAV, facilitating swift and thorough processing and aiding resource-constrained UAVs in conserving their battery power.
[49]	The UAV network is dynamically orchestrated in real-time based on the Quality of Service (QoS) requirements of users through a controller-based UAV swarming approach. The system employs channel switching to automatically deliver customized services to regular and premium users.
[51]	Uses an ant colony algorithm to simulate and analyze the self-organizing behavior exhibited by the swarm robots.
[44]	An algorithm is implemented to choose worker nodes for the execution of diverse nanoservices. The active manager node takes charge of allocating nanoservices to worker nodes that meet their specified requirements, enabling the execution of multiple nanoservices on a single worker node.
[66]	The manager node uses its IP address and port to expose swarm services to all clients. Requests originating from clients are directed to a selected worker node through the internal load-balancing mechanism of the swarm manager, ensuring even distribution of requests.
[68]	The introduction of a global best artificial fish enhances the behaviors of the existing artificial fish, leveraging the collective experience of group members for subsequent actions. To improve results, a time-variant value for step and visual is incorporated, starting with a larger initial value and gradually reducing as the problem approaches its global optimum.
[55]	A distributed computing architecture is established, encompassing an edge/fog/cloud monitoring system and a capillary container orchestrator. This architecture is designed to effectively manage highly dynamic IoT environments where the edge node may experience runtime overloads due to an increase in workload.
[45]	The Priority Task Scheduling (PTS) approach relies on two primary factors: dynamic task allocation (DTA) and resource balancing and availability (RBA).
[63]	The infrastructure is designed to accommodate a wide range of heterogeneous devices, from high-performance Cloud-oriented hardware to smart sensors and actuators at the edge. A workload model enables the description of applications as a collection of components and their relationships, which facilitates the definition of constraints and optimization criteria policies aimed at maximizing the QoS of applications.
[14]	The swarm system consists of smart nodes engaging in communication using ''dumb messages''. The approach reverses the conventional viewpoint on communication through asynchronous messages. The idea is that distributed nodes do not actively communicate but rather are visited by ''families of intelligent messages''.
[13]	The flocking control system comprises two loops. The first loop focuses on controlling the swarm model, generating a trajectory termed the ''warm center''. The second loop is responsible for controlling the UAVs within the swarm, ensuring they track the generated trajectory (or the swarm center) established by the first loop.
[38]	Each drone within the swarm transmits real-time data about plume boundaries, encompassing critical details such as wind direction, speed, location, chemical composition, and air density measurements. The drones dynamically coordinate as a swarm, continuously adapting their positions to trace the outer perimeter of the plume.
[46]	A workflow is created using various Cloud services, and it undergoes dynamic monitoring, adaptation, and self-healing mechanisms to respond to environmental parameters and state changes.
[74]	Multistrategy Ensemble Particle Swarm Optimisation (MEPSO) strives to attain an effective equilibrium between exploration and exploitation by allocating specific segments of its population to handle exploitation tasks while assigning the remainder to focus on exploration endeavors.
[52]	A Multi-Objective Particle Swarm Optimisation (MOPSO) approach is employed, utilizing sequential coding for discrete PSO. The position of each particle serves as a candidate solution to the problem, and the number of tasks aligns with the dimensionality of the particle.
[47]	A Thing Manager periodically polls data from the Thing Directory (TDir) to maintain a comprehensive list of active Servients/WTs along with their Thing Descriptions (TDs). The Optimiser role is responsible for executing the WT/Servient allocation policy. The Migration component receives the deployment plan from the Optimiser and subsequently implements the WT hand-off events as specified in the plan.
[53]	A Kubernetes cluster primarily comprises a control plane and a worker plane. The control plane is typically represented by a single master node, or alternatively, master control components can be distributed across the cluster to enhance fault tolerance.
[62]	Dynamic Yet Another Resource Negotiator (DYARN) incorporates a job scheduler that effectively prioritizes jobs by adjusting the allocation of containers. Jobs with higher priority receive a greater number of containers, thereby reducing the available containers for regular jobs. When the job scheduler selects a job with regular priority, DYARN temporarily decreases the current number of containers needed by the job to either zero or a designated fraction (based on user preferences).
[37]	The Particle Swarm Optimisation (PSO) method is a global minimization technique designed to address problems where the solution is represented by a point or surface within an n-dimensional space. Within this space, particles in the swarm are assigned elementary velocities, and communication channels between particles, akin to the channels of communication between edge nodes, are established.
[56]	A real-time Multi-Agent System encompasses the core elements of MAS, including the agent's internal scheduler, communication middleware, and negotiation protocol, operating in real-time scenarios.
[59]	Edge Infrastructure Pool Framework, which oversees resource expansion within the edge infrastructure. Edge/Cloud Continuum Management Framework which manages resource assignment, availability, security, and more across user sites in the edge/Cloud continuum. And Application Management Framework, which offers integrated DevOps automation for streamlined application lifecycle management.
[42]	The approach comprises three primary functionalities: slot assignment, slot migration, and slot releasing operations. In this method, nodes possess the capability to dynamically organize themselves by being aware of the nearby topology. This not only mitigates the occurrence of unnecessary control packet transmissions but also ensures highly efficient utilization of the channel.
[32]	Backend framework designed for sensor networks and a configurable simulation tool to predict the behavior of manufacturing systems.
[36]	An architecture-level blockchain solution that aligns trust concerns, conceptually captured by the W3C Provenance framework.

TABLE 6. *(Continued.)* **A high-level overview of existing frameworks.**

[34]	In the context of sustainable smart factory objectives, the integration of embodied and situated learning from continuous improvement by lean teams into knowledge engineering processes is pursued, mitigating static manufacturing complexity through self-optimizing, goal-oriented, and self-organizing mechanisms. Operators are considered Cyber-Physical Systems (CPS), incorporating cognitive dimensions and operational learning styles into the system framework.
[64]	Machine-to-Machine (M2M) Framework. It facilitates communication with devices through various M2M protocols, enabling applications and services to access device data in a standardized manner.
[69]	Resource Description Framework (RDF) is employed for low-level knowledge representation and quick reasoning capabilities. Web Ontology Language (OWL) is used for high-level knowledge representation and more intricate reasoning processes. Simple Object Access Protocol (SOAP) is employed to distribute data and knowledge. Web Services Description Language (WSDL) is used to describe web services, while SA-WSDL (Semantic Annotations for WSDL) is employed to provide semantics to web service descriptions. Web Services Business Process Execution Language (WSBPEL) is used for describing workflows.
[41]	Each particle is tasked with resetting its record of the best position to adapt to changes in the environment, preventing decisions based on outdated information. Two methods are considered; Periodic Resetting involving resetting based on the iteration count, where at specific intervals, the particle resets its best position record; and Triggered Resetting in which resetting is triggered by the magnitude of the change in the environment.
[70]	In the Arrowhead Framework, orchestration and configuration systems play a crucial role. The service consumer can query information about a specific service of interest by providing its exact name to the Orchestration System. Subsequently, the Orchestration System retrieves the relevant information from the Service Registry and Authorisation System. It then assembles a list of authorized services along with their corresponding service providers.
[67]	The IoT-Concentrator takes on the responsibility of aiding various IoT sensors/actuators in aggregating data and relaying control for IoT devices. The IoT gateway serves as a mediating gateway between IoT and the Cloud. The IoT gateway can temporarily store collected data and forward it whenever possible, acting as an intermediary between IoT and the Cloud.
[40]	The population is partitioned into several smaller swarms, and there is frequent regrouping of these swarms to facilitate the exchange of information among them. This process persists until a predefined stop criterion is met.
[35]	IDPS for monitoring network intrusions. To efficiently and simultaneously provision security services for multiple devices, containerization is used, and the Docker bridge network is replaced with Open vSwitch (OVS). OVS serves as a virtual router for fog nodes, creating a bridge network with assigned IP addresses for each security service container. The Security Orchestration (SO) system autoconfigures OVS, updating the list of security service containers and their IPs and overcoming scalability and interoperability limitations of the existing Docker bridge network.
[65]	The two-stage optimization method employs IMOPSO (Pareto set of non-dominated solutions) in conjunction with AHP (Best distribution based on application-specific judgment matrix) to achieve the optimal distribution of services.
[39]	A virtual forces model is utilized for controlling the movement of each UAV, leveraging persistent and mutable system parameters. Evolution is governed by eleven strategic parametric mutations, including a neutral benchmark process.
[43]	Eight scalar parameters, influencing the dynamics of swarm behavior in a 3D simulation, are evolved to generate agents capable of collectively flying in line, forming a ring, and creating a figure-eight formation.
[54]	The system enables dynamic orchestration at runtime, ensuring QoS guarantees. Optimization driven by data-driven insights and learning-based tuning becomes instrumental in enhancing orchestration intelligence.
[33]	Aggregate computing shifts the focus from the behavior of individual agents to an aggregate-centric viewpoint, emphasizing the global behavior of a collective or aggregate system. This perspective considers the interactions and behaviors of a whole set of autonomous entities working together.
[71]	A bio-inspired algorithm was deployed to route pallets within the conveyor system. Pallet agents, as they navigate through the conveyors, record their movements. Upon reaching the desired goal, a pallet eliminates duplicated routes and subsequently updates the corresponding pheromone levels on each traversed conveyor.
[50]	A comprehensive framework is established for the Wireless Power Transfer (WPT) charging problem, where a swarm of UAVs benefit by transferring energy to the IoT devices, and concurrently, the charging stations (CSs) benefit from charging the UAVs.
[72]	A dynamic neighborhood PSO is presented. Calculate distances in the fitness space for the current particle's first objective function. Find the m nearest neighbors based on these distances. Identify the local optimum among neighbors using the second objective function's fitness values.
[61]	In the described system, an extension is made to enhance the functionality of existing software components. These components provide a diverse vocabulary for specifying the necessary machinery to perform various behaviors. However, they lack the dynamic expressiveness that software is inherently capable of. The introduced dynamism, observed in the fluid exchange of functionality, proves to be a crucial capability when orchestrating the activities of a swarm of robots.
[57]	Docker Swarm functions as an embedded orchestration tool featuring two distinct roles: workers and managers. Workers are exclusively utilized for hosting containers, whereas managers possess the additional capabilities to terminate, update, deploy, and manage running containers within the cluster. Nodes joining the cluster are categorized based on their position in the architecture (Fog, Edge, or Cloud).
[73]	In each node, a Bundle Installer Service (BIS) is installed. BIS deploys a designated Smart Behaviour within its local environment. Upon receiving an Operator Command or a Trigger, the Behaviour Management Service selects the most suitable Smart Behaviour from the Marketplace. It then identifies the most fitting node/s and instructs the BIS of those nodes to install or update the selected Smart Behaviour.
[19]	The framework consists of several key components: The MDM component provides data workflow observability to the other components. The SWM component manages the scheduling and re-scheduling of application workloads. The PDLC component is central to the orchestration. The NetMA component provides network awareness and ensures secure connectivity across pods.
[20]	The proposed architecture includes the scheduler, controller, knowledge reasoning, perception, and physical plant modules. The manufacturing ontology supports all modules except the physical plant module.

In all cases, we intend to focus on frameworks pointing towards some form of orchestration for the edge and/or IoT nodes.

J. TECHNOLOGY STACK

SRQ11 discusses the technologies that have been employed in the proposed solutions. Technologies are categorized into

front-end and back-end, and for the remaining few, it is unknown. In front-end technologies, robots [51] and Docker's Remote API [55] are discussed. In back-end technologies, the majority of work is done in Docker swarm [57], [60], [66], multi-agent systems frameworks [13], [45], [56], PSO [37], [40], [41], [52], [72], [74] and Kubernetes [19], [53], [63], (see Table 7).

Alam et al. [57] use docker swarm technology on edge devices. These devices are depicted as cyber-physical systems capable of hosting Docker containers. By employing virtualization, these devices are managed locally through gateways and centrally orchestrated in the cloud. In a similar context, Singh et al. [66] discussed how Docker Swarm is employed to effectively manage the workload and service discovery of big data applications. Here, the results demonstrate that escalating workloads in big data applications can be efficiently handled by leveraging microservices within containerized environments. Another widely adopted back-end technology within this research domain includes multi-agent systems, as discussed by Mutlag et al. [45]. In their study, a Critical Healthcare Task Management (CHTM) model is introduced and implemented using an ECG dataset. A resource scheduling model is devised among fog nodes at the fog level, and a multi-agent system is proposed to manage the network comprehensively from the edge to the cloud. They deploy four kinds of agents: personal agent (PA), master personal agent (MPA), fog node agent (FNA), and master fog node agent (MFNA). In the proposed model, three levels of processing are provided–PAs, FNAs, and the cloud–with two levels of control: MPAs and master fog nodes (MFNs). This scheduling strategy claims to guarantee dynamic allocation of tasks and availability of resources.

TABLE 7. Back-end technologies.

Study	Technologies
[32], [57], [60], [66]	Docker Swarm.
[44], [59]	Edge Computing.
[68]	Artificial Fish-Swarm Algorithm.
[13], [45], [56]	Multi-agent system.
[19], [53], [63]	Kubernetes.
[14], [38]	Node.js.
[47]	The Migratable Web of Things (M-WoT) represents an innovative architectural framework designed to facilitate the dynamic allocation of WT to the available computational nodes.
[62]	Hadoop.
[37], [40], [41], [52], [65], [72], [74]	Particle Swarm Optimization.
[36]	Blockchain.
[20], [34]	Cyber-Physical Production System (CPPS) technologies.
[69]	RDF for low-level knowledge representation and fast reasoning, OWL for high-level knowledge representation and more complex reasoning, SOAP for data and knowledge distribution, WSDL for describing Web services, SA-WSDL for describing the semantics of Web services, WSBPEL for describing workflows.
[70]	Arrowhead Framework version 4.0.0, Jena TDB triplestore and its API.
[35], [64]	OpenFog architecture - a way to implement orchestration in addition to security assurance.
[33]	Edge servers, MQTT (Message Queuing Telemetry Transport), low-power/long-range communication via LoRaWAN, and Cloud offloading.
[73]	Paremus Service Fabric (OSGi-Open Standard for Software Modularity R7 compliant platform).

K. SEMANTICS CHARACTERISTICS

SRQ12 and **SRQ13** address if semantics is enabled in the dynamic swarm orchestration and what are their involvement, respectively. Out of all the primary studies, only seven studies cover semantic-based behaviors. It is illustrated in Figure 11.

Kuru et al. [48] deployed multiple Leader Fully Automated UAVs (LFAUAVs) for collaborative semantic understanding of both ground and aerial statics and dynamics. This enables the realization of various multi-agent tasks safely and optimally, including the reconstruction of 3D city models with geometric and semantic information. Moreover, [69] uses semantic architecture to perform components such as Global Robot Knowledge Base, Local Robot Knowledge Base, Communication Gateway, Web Services Ecosystem, and Web Services Gateway.

Reference [33] addresses the semantics of an entire logical device, referred to as a node, regarding the behaviors and interactions of its logical components. These components are linked to node identifiers, forming situated components. Wan et al. [20] used the manufacturing ontology to model the abstraction concepts, which include the product, process, operation, and resource, for the manufacturing domain. The context ontology formalizes situational knowledge. Unlike the manufacturing scene, it encompasses the temporal nature of dynamic changes.

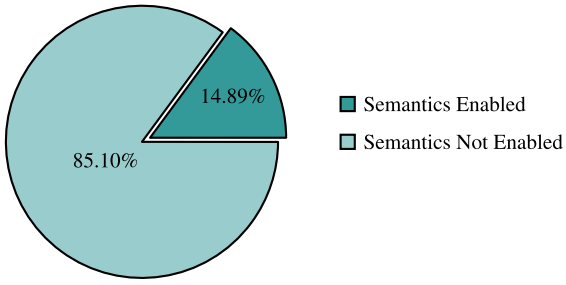

FIGURE 11. Semantic enablement.

L. ADVANTAGES OF THE PRIMARY STUDIES

SRQ14 describes the advantages of the primary studies, which are collected in Table 8. Automation and dynamicity are among the common advantages in the studies, *e.g.*, dynamic resource planning is discussed in [58] and [62] covers orchestration of the multilevel workflow that is augmented with SWSD, automatically.

TABLE 8. Overview of advantages.

Study	Advantages
[60]	Substitutes the manually adjusted supply of cloud services with an automatically adjusted supply.
[58]	Automatically orchestrates the multilevel workflow enhanced with SWSD.
[48]	UAVs enhance their cooperation within their complex ecosystem through mission-oriented swarm intelligence, eliminating constraints related to battery, coverage, and resources within smart cities.
[49]	Can fulfill the data rate requirements of premium and regular users in both simple and complex scenarios.
[44]	Guarantees scalability, resource efficiency, and fault tolerance for highly dynamic resource-constrained IoT scenarios.
[66]	Load balancing and service discovery in Microservices are effectively managed through a Docker Swarm orchestrator.
[68]	Simulation on temperature control of dyeing machine is to compare with the conventional fuzzy logic controller (FLC) and the general artificial fish swarm algorithm (AFSA) based SOFLC, improving accuracy and convergence speed.
[55]	Efficiently handles highly dynamic IoT environments, addressing situations where the edge node may encounter runtime overload due to an increased workload.
[45]	The scheduling strategy manages to dynamically perform task allocation, resource availability, and balancing.
[63]	The scalable orchestration architecture employs a distributed approach for scheduling tasks, diverging from the prevalent centralized methods in current architectures.
[14]	Enables integration between heterogeneous components using swarm systems.
[38]	The swarm of drones system gathers sensor readings for subsequent assessment and offers a visual indicator that aids first responders in visually tracking plume movement over time.
[46]	An integrated and comprehensive architecture is proposed, encompassing IoT workflow specification, orchestration, monitoring, prediction, and adaptation, being mindful of resources and cost considerations.
[52]	Takes into account resource choice for complex applications where multiple tasks are intricately combined within a specific business process. In contrast, others predominantly concentrate on a single service or offload destination.
[53]	A distributed application model with a single compute cluster and a dependable data link is introduced. This unification extends to merging the edge cloud within the public network.
[62]	Dynamic resource planning.
[56]	Mapped MAS key elements to real-time aspects.
[42]	Self-organising design addresses changes in topology and link states without a centralized controller in aerial swarm communication.
[32]	Involves multilevel auto-scaling, covering both virtual machines (VMs) and containers, utilizing a Cloud-agnostic approach.
[34]	Proposes a re-engineering of cognitive manufacturing- between the concept of CPS and the holonic paradigm.
[64]	Orchestrated in the IoT Testbed, services in distributed Fog Nodes, verifying the functional aspects of the architecture.
[69]	Standardised and generic communication paradigm that provides loose coupling between heterogeneous autonomous robotic entities, semantic integration between robots and external environment, and service discovery .
[41]	Adopts dynamic goals if the rate of change of the goal does not exceed the maximum velocity of the swarm.
[70]	Design of a self-adaptation service within a localized automation Cloud. This service orchestrates and configures software systems and devices securely and automatically.
[67]	Implemented container orchestration using Docker Swarm and validated its feasibility for continuous service operations.
[40]	It demonstrates superior performance on intricate multi-modal problems in comparison to certain other PSO variants.
[35]	Optimises edge infrastructure, provisioning security, and other types of services launched for a remote work-site.
[65]	Enables the evaluation of diverse criteria with varying units of measurement, whether qualitative or quantitative.
[39]	Employs an online evolutionary approach, considering the dynamic nature of the environment.
[54]	Outlines challenges in developing an orchestration framework that spans across all layers within the Fog resource stack.
[33]	Introduced a novel model for self-organizing cyber-physical systems.
[72]	The PSO method offers the advantage of being easy to implement and requiring only a few parameters to be adjusted.
[57]	Facilitates distributed deployments by using a modular architecture based on lightweight virtualization orchestrated by Docker.
[51]	Enables selection of a suitable motion model tailored to specific functions within group robot formation control.
[74]	Improves algorithm convergence by expanding the particle population's search area, preventing entrapment in local optima, and enhancing adaptability to dynamic environmental changes.
[47]	Simultaneously maximizes WoT data locality and balances the computational workload across resources.
[71]	Explores the synergy of multi-agent and holonic systems paradigms, along with service-oriented architectures.
[50]	Matching algorithm attains nearly optimal energy coverage performance and demonstrates superior fairness among operators' profits when compared to alternative schemes.
[61]	Automated concurrency and distributed execution.
[43]	The system can reduce the need for manual tuning of control parameters, offering a practical approach to designing swarm systems through interactive genetic programming.
[59]	Manages a Cloud/Edge federation's resource pool by recruiting edge devices. Enables the execution of NextGen applications with ultra-low latency requirements.
[37]	Optimises processing, propagation delays, and end-to-end latency by employing a controller and utilizing a PSO algorithm.
[36]	Blockchain-based solution effectively aligns trust concerns at an architectural level using the W3C Provenance framework.
[73]	Facilitates the dynamic deployment, orchestration, and monitoring of distributed applications. Automatically install new behaviors in response to environmental triggers and user events.
[13]	Shows that strong flocking performance relies on careful selection of weighting matrices, PD controller, and swarm parameters.
[19]	Highlights the integration of context-awareness into Edge-Cloud orchestration to enhance resource management.
[20]	Enhances the adaptability of production systems.

M. LIMITATIONS AND COUNTERMEASURES

SRQ15 and **SRQ16** describe the limitations and countermeasures provided in the primary studies. In most of the studies, the limitations/drawbacks are not stated. Only sixteen studies have addressed the limitations (described in Table 9).

N. FUTURE DIRECTIONS

Regarding **SRQ17**, providing a heterogeneous outlook on future challenges appears somewhat disparate. However, on a more general level, these challenges can be broadly classified into three distinct categories as the following:

1) Functionality-related challenges pertain to the integration of novel functionalities to be implemented (Table 10).

2) System-related challenges are associated with expanding pre-existing functionalities (Table 11).

3) User-related challenges involve the gathering of user experiences (Table 12).

Figure 12 shows a breakdown of these categories.

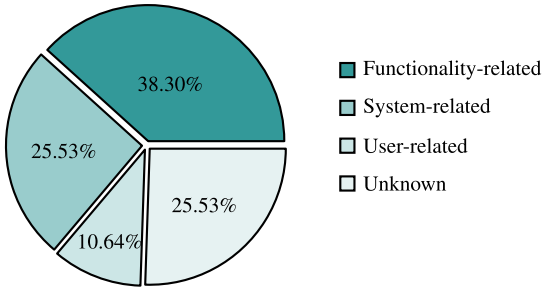

FIGURE 12. Future challenges categories.

Functionality-related challenges are the majority among the primary studies. These include exploring ant colony optimization [37] and further studying how to achieve optimization in distributed and decentralized solutions [71]. Using a ML approach to enable dynamic load and performance-aware service deployment, which would further improve the performance of decentralized nanoservice architectures [44], is also another example of functionality-related challenges.

Some of the instances of system-related challenges cover extending the present edge computing design to include blockchain technologies [55], to check system's fault tolerance in the case of errors and implementing security mechanisms for messages [14] to analyze possible overhead related to control, communication, scheduling, and negotiation [56] and incorporate workflow-driven automation to enable more dynamic service development and operation [67].

Some of the user-related challenges identified are user mobility and enhancements in the model to process all essential signs [45] and to examine the computational complexity and cost efficiency of the proposed work [66].

V. DISCUSSION

This section elaborates on the results produced by the 17 structured research questions, highlighting key takeaways and new perspectives.

The affiliation of the primary studies' first authors are based principally in Asia, North America, and Europe—see Figure 3. The highest concentration of primary studies is in Europe. This could be (partially) explained by the fact that Europe is part of the Digital Decade Policy programme[6] and investing considerably in Cloud, Edge, and IoT research. The primary objective in researching the combined domain of cloud, edge, and IoT is to establish European supply and value chains seamlessly across the continuum. This requires integrating essential components such as computing, connectivity, IoT, AI, and cybersecurity, with a special emphasis on exploring decentralized intelligence concepts like swarm computing.

The majority of application domains focus on IoT, UAVs, optimization, and healthcare—see Figure 4. Recent studies on UAVs utilize swarm-based approaches for their dynamic functionalities. Moreover, optimization-related studies such as PSO and other evolutionary algorithms are becoming popular. Smart healthcare has also been in demand in recent years to elevate the quality of healthcare services for patients. The healthcare domain is introducing IoT devices to be coordinated increasingly (*i.e.*, wearable sensors and home monitoring sensors [44], [45]). Moreover, some studies suggest (arguably) compound IoT devices with Distributed Ledger Technology (DLT) to establish trust between various nanoservice providers, ensuring ample privacy and security for local IoT services [44].

The majority of studies operate within the context of UAVs users, end users, and mobile users. Indeed, such an interest is also confirmed by the U.S. Government Accountability Office (GAO),[7] which reports a spike in interest in drone swarm technologies, employing algorithms and local sensors to synchronize drones with minimal human involvement. Furthermore, the GAO report explains that the progress in AI and drone components has enabled the realization of drone swarms. However, as of today, they are limited to simpler missions such as aerial light shows. The GAO report[8] (aligned with current research, see [77], [78]) also suggests increasing interest in UAV fighting wildfires and finding missing persons, justifying why most users in our primary studies are UAV users.

Several studies concentrated solely on establishing theoretical foundations (Section IV-D) that can support/improve the large number of embryonic applications presented as prototypes. Indeed the primary studies proposing prototypes outnumber conceptual and theoretical primary studies, see Figure 5. There is a range of frameworks among the primary studies, such as multi-objective PSO, M2M Framework, RDF for low-level knowledge representation and fast reasoning, etc., and mostly tested with prototypes, which are not yet fully implemented and/or deployed. Such circumstances imply that most of these technologies are in their initial development

[6]https://digital-strategy.ec.europa.eu/en/policies/iot-investing
[7]https://www.gao.gov
[8]https://www.gao.gov/assets/gao-23-106930.pdf

TABLE 9. Limitations and Solutions.

Study	Limitations	Solutions
[60]	The current implementation represents the initial prototype, which is capable of managing only basic alerts and operating according to straightforward policies.	Solution not provided.
[34]	Did not address cybersecurity.	Solution not provided.
[38]	To find out how the plume is moving vertically along the z-axis. To find out how the drones are affecting the plume, particularly how much the swarm is moving contaminated air downwards towards first responders.	In future work, the plan is to overcome these limitations by enhancing the degrees of freedom in flight movements and expanding the swarm's size to offer additional cross-sections along the z-axis.
[62]	There is no restriction on the quantity or percentage of jobs that can be executed with high priority.	In the future, the system administrator will function to set the limitation.
[64]	The assessment lacks an evaluation of the non-functional aspects of the implementation.	Solution not provided.
[72]	The discussion here focuses solely on the initial step in investigating the resolution of multiobjective parameter optimization problems using particle swarm optimization.	Solution not provided.
[13]	The assumption made is that the dynamic characteristics of agents are uniformly distributed.	Solution not provided.
[71]	Optimality and context adaptability are not guaranteed.	Solution not provided.
[55]	The operational conditions, encompassing factors like time-varying processing delays, CPU and I/O load factors, along with the quality of QoS attributes of Edge, Fog, or Cloud infrastructures, such as availability, may exhibit runtime variations. These variations occur independently of the workload characteristics and geographic location of IoT devices.	In the future, running infrastructures will be consistently assessed based on the specified characteristics.
[69]	Scalability of rule engine and efficient service discovery algorithms.	Solution not provided.
[70]	Utilizing Semantic Web Rule Language (SWRL) for articulating high-level adaptation policies may pose challenges, as this approach can be error-prone and challenging for domain experts to specify and verify.	A tool will be developed to support users in creating, editing, and validating adaptation policies.
[51]	Failure to account for the size and turning radius of the wheeled robot has led to an uneven distribution once the capture process is finalized.	The distribution of the robot will be improved in their next step.
[35]	The orchestrator's capabilities depend on the nature of the service type (services that can be launched for a remote work site).	Develop and validate the other orchestrating strategies in future work.
[41]	To examine the effect of distributing the reset operation over some number of iterations rather than abruptly cleansing the memories of all particles simultaneously. To determine a method of detecting a change in the environment at the particle level rather than using an outside trigger. To evaluate the effect of the changing environment on the choice of the inertia parameter.	Solution not provided.
[19]	Convergence times in federated environments need to be tested, and an analysis of the solver's waiting times to deploy workloads should be conducted under different conditions.	Solution not provided.
[20]	Assumes that the control parameters governing machine behavior are accessible and that any machine failures can be rectified by adjusting these parameters.	This limitation can be overcome through the continuous advancement and implementation of Cyber-Physical Production Systems (CPPS).

phases, opening for remarkable research opportunities, although not tackling yet real-world challenges nor dealing with their constraints.

Among the few real-world tested applications attempting to tackle dynamic orchestration, it is worth mentioning the coordination of UAVs as a swarm to dynamically align their position to follow the outer edges of the plume in air-fire fighting [38]. The UAVs swarm is intended to assist the first responders in identifying the plume movement and predicting and isolating the impact area. However, in this prototype test, only one direction (horizontal) movement of the drone swarm is explored. It is inferred from the SLR analysis that within a single swarm, it is also useful to note the categories of the nodes beforehand, *e.g.*, manager and worker nodes, swarm controller and swarm center, master and slave nodes, and worker and balancer nodes. These roles-based concepts assist in better orchestration and provide dynamicity as each of the nodes, depending on their role, knows their next action plan, *e.g.*, to stay, roles to perform, or leave the swarm. However, determining the number of manager nodes to implement is a trade-off between performance and fault-tolerance considerations. Increasing the number of manager nodes enhances the fault tolerance of the swarm. However, the addition of extra manager nodes diminishes write performance due to the increased number of nodes required to acknowledge proposals for updating the swarm state, which leads to higher network round-trip traffic. Additionally, there might be a situation of single-point-of-failure (*e.g.*, if the manager or master nodes are damaged/hacked), which could break the whole swarm orchestration, and manual intervention might be needed to fix it. To eliminate such situations, redundancy should be added to the system [79].

Besides single swarm and their nodes' roles, the multi-swarm feature is also discussed in the primary studies (*i.e.* [40]). Here, the population is divided into small

TABLE 10. Functionality-related future challenges.

Study	Future Challenges
[58]	Introducing fault-tolerant and context-aware composition is the objective, with a focus on leveraging state-of-the-art bio-inspired algorithms like ant colony optimization (ACO), evolutionary algorithm (EA), and particle swarm optimization (PSO) for the refinement of the web service composition process.
[48]	To formulate approaches for FAUAVs, with a specific focus on positioning a swarm in aerial space safely and efficiently. The goal is to minimize interferences, taking into account the individualized tasks assigned to each FAUAV.
[44]	Leveraging AI/ML methodologies to facilitate dynamic load and performance-aware service deployment holds the potential to enhance the efficiency of decentralized nanoservice architectures.
[46]	Cognitive computing.
[47]	Employing ML techniques for the seamless and adaptive deployment of WTs within distributed WoT environments is a key focus of this investigation.
[53]	To present a framework designed to optimize and effectively distribute diverse traffic types, catering to a range of applications that demand swift server responses.
[37]	Ant colony optimization or firefly algorithm.
[36]	Off-chain storage and other blockchain configurations
[34]	Cyber security, virtual and augmented reality, additive manufacturing, collaborative robotics, RFID, M2M or wearables, and Cloud enablers such as big data, CMfg, or IoT could.
[64]	To focus on measuring orchestration time and system behavior when orchestrating diverse services under varied setups, such as different image sizes and locations. It evaluates opportunity losses by verifying successful orchestration with all requirements met simultaneously. The investigation will also address the time needed for scaling and migrating both individual microservices and the entire service.
[69]	Scalability of Rule engine, WSG, and efficient service discovery algorithms.
[70]	To consider the integration of ML and Reinforcement Learning into the MAPE-K model.
[35]	To develop and validate the other orchestrating strategies.
[39]	To investigate bridging evolutionary learning heuristics with existing machine learning architectures, specifically within AI framework development.
[33]	To examine and develop mechanisms facilitating the adaptive and opportunistic deployment of self-organizing systems, with careful consideration given to the dynamics present in the environment.
[71]	To investigate methods for achieving optimization in distributed and decentralized solutions.
[19]	To evaluate various methods for combining metrics tailored to specific target profiles.
[20]	To focus on developing more effective rescheduling strategies and algorithms.

TABLE 11. System-related future challenges.

Study	Future Challenges
[60]	To enhance scalability for more intricate optimization scenarios and incorporate user policy enforcement, such as orchestrating applications based on security policy specifications.
[55]	To extend the present Edge computing design to include blockchain technologies.
[14]	To provide the system's fault tolerance in the case of errors and implement security mechanisms for messages.
[38]	To augment the degrees of freedom in flight movements and enlarge the swarm size to introduce additional cross-sections along the z-axis.
[56]	To assess potential overhead associated with control, communication, scheduling, and negotiation processes.
[59]	To unveil a comprehensive concrete architecture of the system, presenting the initial integrated version of the ACCORDION Platform.
[67]	To integrate workflow-driven automation to facilitate dynamic service development and operation.
[65]	To employ it within an actual orchestrator of IoT infrastructure to practically assess how various placements of services within fog nodes impact the overall performance of the IoT system.
[43]	To advance from static control parameter lists to the evolution of actual simulation code for a more dynamic and adaptive approach.
[54]	To enhance the parallelization of the simulation and optimize the complexity of GA-Par to achieve improved scalability over large-scale infrastructures.
[72]	To scrutinize parameters and their impact on optimization performance, particularly in the current PSO version, to examine the treatment of constraints in the PSO algorithm, initially designed for constraint-free multiobjective optimization, and to investigate and compare the PSO algorithm with other evolutionary approaches, evaluating their effectiveness across different optimization scenarios.
[73]	To develop functional elements for target use cases; BRAIN-IoT will intensify cascading failure events to assess its mean-time-to-recovery capabilities.

TABLE 12. User-related future challenges.

Study	Future Challenges
[66]	To assess and address the computational complexity and cost-effectiveness inherent in the proposed work.
[45]	User mobility and improvements in the model for processing all vital signals.
[62]	Let the system administrator set the limitation.
[42]	To test the protocol in real-world operations, which may have an impact on the quality of evaluations. More detailed network scenarios could be examined to observe the recovery process once a collision occurs.
[41]	To evaluate the effect of the changing environment on the choice of the inertia parameter.

swarms that frequently regroup to exchange information. The neighborhood structure, combining exploitation and exploration, outperforms other PSO variants on complex multi-modal problems. Indeed, tackling the multi-swarm concept can bring great benefits to studies such as [38]. For example, elaborating on a possible extension of such a study, we could consider a swam (swarm-1) of UAVs to deliver a shipping package, and swarm-2 already has the information about the firefighting region. In this situation, once the fire region is identified by swarm-2, taking advantage of the multi-swarm concept, swarm-2 dynamically could exchange these fire-affected region's details with swarm-1. It would then allow swarm-1 to avoid these regions to deliver the packages. Hence, in practical situations, it is imperative to

recognize the significance of multi-swarm configurations and the exchange of information among swarms for mutual awareness.

Taking real-world scenarios into consideration, the requirements of the primary studies are also identified. They were classified into three categories, namely, functional, architectural, and front-end, with functional requirements outnumbered in the primary studies. Indeed, most of the functional requirements consist of dynamic orchestration [62], [63], dynamic scalability [14], [60] and resource discovery [44] which are exceptionally crucial in practical, real-world scenarios. With the rise of IoT devices, it is extremely critical to orchestrate, scale, and perform resource discovery on the fly to regulate their interactions seamlessly, *i.e.*, demonstrating dynamicity, which is discussed in all the primary studies and is validated by the recent survey conducted by Firouzi et al. [80]. Additionally, most of the advantages observed in the primary studies are replacing manual work and providing dynamic resource planning. These, in turn, contribute to cost reduction and facilitate efficient resource planning of IoT devices.

One of the ways to implement the dynamicity feature is through microservices, which were also identified as the most common services provided by the primary studies. Microservices simplify complexity through small applications, adaptability in crafting diverse structures and tools, reliable scalability, and enhancing the fault tolerance of the service. However, there still are ongoing challenges in the IoT security field related to microservices (see [81]). To summarise the methodologies and the key features in the primary studies (Dynamicity (D), Edge computing (EC), Orchestration (O), Semantics (S), Swarm-intelligence (SI)), a comparative table is shown in Table 13. Certain studies encompass more key features than others, which is indicated by highlighting these studies in brackets. For example, all swarm-based approaches exhibit the key features of dynamicity, orchestration, and swarm-intelligence, with [57] additionally covering the edge computing feature; this is represented as D, EC ([57]), O, SI.

Technologies that have been employed in the primary studies are categorized into back-end and front-end. In the back-end technologies, most work is done in Docker swarm and multi-agent systems. However, according to the recent survey by Cilic et al. [82], the authors concluded that Docker Swarm is designed with a primary focus on simplicity but is not specifically tailored for handling distributed workloads. It is frequently presented as a streamlined substitute for Kubernetes, lacking sophisticated automation capabilities. Moreover, Calvaresi et al. [56] discuss the challenge of real-time multi-agent systems for enabling IoT. They have discussed the recommended characteristics of these systems should be intelligence, autonomy, and real-time behavior. Still, off-the-shelf MAS mainly addresses only the first two characteristics and fails to comply with strict timing constraints. Overall, the current leading-edge technologies are not applicable to real-time scenarios due to the above limitations.

Recently, with heterogeneous data in the real world, semantics characteristics have become a growing topic in literature for creating interoperable IoT applications. Semantics create abstractions that capture the essential capabilities, goals, roles, and tasks performed by IoT edge devices [16]. It is inferred from the aggregation of the SLR analysis that, given these capabilities in advance, IoT devices become platform-independent and format-neutral, and IoT applications can be reused, ultimately reducing the cost and time associated with application development. However, only seven out of the primary studies cover semantic-based behaviors, implying abundant opportunities for semantics within this field. Moreover, it is envisioned that given an IoT platform with semantics-enabled characteristics, there is a possibility to add an orchestrator and coordinator to orchestrate IoT edge devices, *i.e.*, an orchestrator to coordinate the swarms and a coordinator to get nodes to join/leave the swarm. If semantics describing the nodes and swarm are already in place, the orchestrator could use this to gain an advantage in getting information and coordinating inter-swarm interactions. Similarly, a coordinator could utilize the semantic information about the roles and tasks of the nodes and assign/manage them accordingly.

In [38], the current limitation is that only the plume along the z-axis is monitored. However, to foster real-world adoptions, it is necessary to extend the coordination to multiple directions. Similarly, limitations and their recommended solutions are identified in the primary studies (Table 9). Unfortunately, only sixteen studies out of the primary studies have explicitly stated their limitations. A limited number of studies (43.75%) had specified as their countermeasures that they would work on their limitations in their future work. Still, most of the drawbacks mentioned in the primary studies (64.29%) did not provide any solution. For instance, Imrith et al. [35] have stated their limitation that the orchestrator's capabilities vary based on the service type. Here, their solution is to develop and validate the other orchestrating strategies in their future work. Drawing inferences from the SLR results, these strategies have to be dynamic to tackle real-world problems, such as adapting to diverse service types and devices and transitioning when a device depletes its battery. Additionally, these limitations could be a good starting point for young researchers to delve into this line of research.

A possible approach is discussed in the recent studies [83]. Here, the author is looking to build smart applications that work together, like a swarm of bees. The intended plan is to use ready-made templates for these applications and connect them to devices that are available. The aim is to describe what these applications should do in a simple way using models that computers can understand. Essentially, using clear and easy-to-understand instructions, these templates will define the business needs. Then, these templates can be quickly

TABLE 13. Methodolgies and key features (Dynamicity (D), Edge Computing (EC), Orchestration (O), Semantics (S), Swarm-intelligence (SI)).

Studies	Methodologies	Key Features
[33], [34], [48], [56]	Agent-Based Architecture	D, EC([33], [34], [48]), O, S([33], [48]), SI([48])
[51], [68], [71]	Bio-inspired Algorithm	D, O, SI
[46]	Cloud-service Orchestration	D, EC, O, SI
[54], [64], [73]	Fog Orchestration	D, EC, O, SI([54], [64])
[19], [32], [53], [55], [60]	Microservices-based Orchestrator	D, EC ([19], [53], [55]), O, S([19]), SI([32], [60])
[44]	Nanoservices-based Approach	D, EC, O, SI
[59], [63]	Orchestration for cloud-edge continuum	D, EC, O
[37], [41], [52], [65], [72], [74]	Particle Swarm Optimisation	D, EC([37], [65]), O, SI
[45], [62]	Priority Scheduling Approach	D, EC, O
[35], [36]	Security Orchestrator	D, EC, O, SI([36])
[20], [47], [58], [69], [70]	Semantics-based Approach	D, EC([47], O, S, SI([47], [69])
[13], [14], [38], [39], [40], [42], [43], [49], [50], [57], [61], [66], [67]	Swarm-based Approach	D, EC ([57]), O, SI

turned into smart applications that work together. This will make it easy to connect these applications to devices and set them up to achieve new business goals. Based on our findings, the following potential research questions can be outlined: Can semantic models be used to represent interactions among IoT edge devices? Can we use these semantic descriptions to empower and enable dynamic swarm orchestration among the nodes? Can we use swarm intelligence for optimizing the orchestration of tasks of edge devices, cooperating towards a common goal?

Finally, future directions covering functionality-related, system-related, and user-related challenges are discussed in Table 10, Table 11, and Table 12, respectively. Functionality-related works (*e.g.*, ant colony optimization, machine learning approach, etc.) are the majority among primary studies, implying novel functionalities are encouraged to be explored. However, pros and cons have to be weighed in advance before implementing these future work, *e.g.*, machine learning or deep learning may introduce a new set of issues (no-one-size-fit-all, longer convergence time, butterfly effect, etc.) supported by the studies [84]. Similarly, blockchain configurations can be incorporated in future work for [36]. Nevertheless, it is to be noted the issues related to the IoT-blockchain integration (See [85]) and if blockchain is suitable for this type of data in order to comply with the privacy regulations (*e.g.*, European Union's General Data Protection Regulation - GDPR) (See [86]). When dealing with people, ensuring their control over data is paramount. The introduction of more stringent data privacy regulations *e.g.*, GDPR emphasizes the necessity for next-generation systems to address this issue comprehensively, with no tolerance for oversight.

VI. CONCLUSION

This paper presents an SLR of 49 primary studies that highlight the contributions in the areas of dynamic swarm orchestration and the use of semantics in edge nodes. The contribution of the paper is three-fold: (i) To enhance understanding of the motivations and relevance of existing contributions in dynamic swarm orchestration and semantics in edge nodes, it conducts a SLR of the current state of the art. This review captures the different demographic and application domains, intended user classes, goals, requirements, scope granularity and dynamicity, services, frameworks, technology, semantic capabilities, and advantages. (ii) To formalize the open challenges associated with applying dynamic swarm orchestration and semantics in edge nodes and offer directions for future research. (iii) To present key findings for future research which are drawn inferences from the SLR results.

Specifically, it follows a rigorously established methodology distinguished by seventeen structured research questions. The SLR explains the review planning, including query selection, inclusion and exclusion criteria, and resolution policies. An in-depth per-feature analysis of the identified aspect has been conducted, and the collective insights have been consolidated in a comprehensive discussion. The key findings highlighted in the papers are as follows: (i) a considerable number of research papers present conceptual inquiries, often lacking evaluations or focusing on relatively uncomplicated scenarios; (ii) there is a scarcity of efforts dedicated to exploring semantics in this field; (iii) merely around half of the primary studies delve into swarm or multi-swarm-based approaches. This represents a noteworthy opportunity for future research and development in this field of semantics-based dynamic swarm orchestration in IoT edge devices. The insights drawn from this study can be valuable for both theoretical and practical dimensions in future research initiatives.

REFERENCES

[1] M. A. Ahad, S. Paiva, G. Tripathi, and N. Feroz, "Enabling technologies and sustainable smart cities," *Sustain. Cities Soc.*, vol. 61, Oct. 2020, Art. no. 102301.
[2] A. Kirimtat, O. Krejcar, A. Kertesz, and M. F. Tasgetiren, "Future trends and current state of smart city concepts: A survey," *IEEE Access*, vol. 8, pp. 86448–86467, 2020.
[3] W. Li, T. Logenthiran, V.-T. Phan, and W. L. Woo, "A novel smart energy theft system (SETS) for IoT-based smart home," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 5531–5539, Jun. 2019.
[4] C. K. Metallidou, K. E. Psannis, and E. A. Egyptiadou, "Energy efficiency in smart buildings: IoT approaches," *IEEE Access*, vol. 8, pp. 63679–63699, 2020.

[5] S. Ramalingam, K. Baskaran, and D. Kalaiarasan, "IoT enabled smart industrial pollution monitoring and control system using raspberry pi with BLYNK server," in *Proc. Int. Conf. Commun. Electron. Syst. (ICCES)*, Jul. 2019, pp. 2030–2034.
[6] A. H. Sodhro, M. S. Obaidat, Q. H. Abbasi, P. Pace, S. Pirbhulal, A.-U.-H. Yasar, G. Fortino, M. A. Imran, and M. Qaraqe, "Quality of service optimization in an IoT-driven intelligent transportation system," *IEEE Wireless Commun.*, vol. 26, no. 6, pp. 10–17, Dec. 2019.
[7] I. Rodriguez, R. S. Mogensen, A. Schjørring, M. Razzaghpour, R. Maldonado, G. Berardinelli, R. Adeogun, P. H. Christensen, P. Mogensen, O. Madsen, C. Møller, G. Pocovi, T. Kolding, C. Rosa, B. Jørgensen, and S. Barbera, "5G swarm production: Advanced industrial manufacturing concepts enabled by wireless automation," *IEEE Commun. Mag.*, vol. 59, no. 1, pp. 48–54, Jan. 2021.
[8] N. Mishra and S. Pandya, "Internet of Things applications, security challenges, attacks, intrusion detection, and future visions: A systematic review," *IEEE Access*, vol. 9, pp. 59353–59377, 2021.
[9] K. Cao, Y. Liu, G. Meng, and Q. Sun, "An overview on edge computing research," *IEEE Access*, vol. 8, pp. 85714–85728, 2020.
[10] K. Kaur, S. Garg, G. S. Aujla, N. Kumar, J. J. P. C. Rodrigues, and M. Guizani, "Edge computing in the industrial Internet of Things environment: Software-defined-networks-based edge-cloud interplay," *IEEE Commun. Mag.*, vol. 56, no. 2, pp. 44–51, Feb. 2018.
[11] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: A survey," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 450–465, Feb. 2018.
[12] H. Singh, A. Bhasin, and P. R. Kaveri, "QRAS: Efficient resource allocation for task scheduling in cloud computing," *Social Netw. Appl. Sci.*, vol. 3, no. 4, pp. 1–7, Apr. 2021.
[13] E. Joelianto and A. Sagala, "Swarm tracking control for flocking of a multi-agent system," in *Proc. IEEE Conf. Control, Syst. Ind. Informat.*, Sep. 2012, pp. 75–80.
[14] L. Alboaie, S. Alboaie, and A. Panu, "Swarm communication—A messaging pattern proposal for dynamic scalability in cloud," in *Proc. IEEE 10th Int. Conf. High Perform. Comput. Commun., IEEE Int. Conf. Embedded Ubiquitous Comput.*, Nov. 2013, pp. 1930–1937.
[15] Y. Miao, K. Hwang, D. Wu, Y. Hao, and M. Chen, "Drone swarm path planning for mobile edge computing in industrial Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 19, no. 5, pp. 6836–6848, May 2023.
[16] A. S. Thuluva, D. Anicic, S. Rudolph, and M. Adikari, "Semantic node-RED for rapid development of interoperable industrial IoT applications," *Semantic Web*, vol. 11, no. 6, pp. 949–975, Oct. 2020.
[17] L. D. Ngan and R. Kanagasabai, "Semantic web service discovery: State-of-the-art and research challenges," *Pers. Ubiquitous Comput.*, vol. 17, no. 8, pp. 1741–1752, Dec. 2013.
[18] P. C. Calcina-Ccori, L. C. C. De Biase, G. Fedrecheski, F. S. C. da Silva, and M. K. Zuffo, "Enabling semantic discovery in the swarm," *IEEE Trans. Consum. Electron.*, vol. 65, no. 1, pp. 57–63, Feb. 2019.
[19] R. C. Sofia, J. Salomon, S. Ferlin-Reiter, L. Garcés-Erice, P. Urbanetz, H. Mueller, R. Touma, A. Espinosa, L. M. Contreras, V. Theodorou, and N. Psaromanolakis, "A framework for cognitive, decentralized container orchestration," *IEEE Access*, vol. 12, pp. 79978–80008, 2024.
[20] G. Wan, X. Dong, Q. Dong, Y. He, and P. Zeng, "Context-aware scheduling and control architecture for cyber-physical production systems," *J. Manuf. Syst.*, vol. 62, pp. 550–560, Jan. 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0278612522000097
[21] M. Razian, M. Fathian, R. Bahsoon, A. N. Toosi, and R. Buyya, "Service composition in dynamic environments: A systematic review and future directions," *J. Syst. Softw.*, vol. 188, Jun. 2022, Art. no. 111290.
[22] R. C. Sofia, D. Dykeman, P. Urbanetz, A. Galal, and D. A. Dave, "Dynamic, context-aware cross-layer orchestration of containerized applications," *IEEE Access*, vol. 11, pp. 93129–93150, 2023.
[23] E. Gagliardi, G. Bernardini, E. Quagliarini, M. Schumacher, and D. Calvaresi, "Characterization and future perspectives of virtual reality evacuation drills for safe built environments: A systematic literature review," *Saf. Sci.*, vol. 163, Jul. 2023, Art. no. 106141.
[24] D. Calvaresi, A. Dubovitskaya, J. P. Calbimonte, K. Taveter, and M. Schumacher, "Multi-agent systems and blockchain: Results from a systematic literature review," in *Advances in Practical Applications of Agents, Multi-Agent Systems, and Complexity: The PAAMS Collection: 16th International Conference, PAAMS 2018, Toledo, Spain, June 20–22, 2018, Proceedings 16*. Cham, Switzerland: Springer, 2018, pp. 110–126.
[25] B. Kitchenham, O. P. Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman, "Systematic literature reviews in software engineering—A systematic literature review," *Inf. Softw. Technol.*, vol. 51, no. 1, pp. 7–15, Jan. 2009.
[26] Y. Mualla, A. Najjar, A. Daoud, S. Galland, C. Nicolle, A.-U.-H. Yasar, and E. Shakshuki, "Agent-based simulation of unmanned aerial vehicles in civilian applications: A systematic literature review and research directions," *Future Gener. Comput. Syst.*, vol. 100, pp. 344–364, Nov. 2019.
[27] S. Anjomshoae, A. Najjar, D. Calvaresi, and K. Främling, "Explainable agents and robots: Results from a systematic literature review," in *Proc. 18th Int. Conf. Auto. Agents Multiagent Syst. (AAMAS)*, Montreal, QC, Canada, May 2019, pp. 1078–1088.
[28] M. Galster, D. Weyns, D. Tofan, B. Michalik, and P. Avgeriou, "Variability in software systems—A systematic literature review," *IEEE Trans. Softw. Eng.*, vol. 40, no. 3, pp. 282–306, Mar. 2014.
[29] T. Tahir, G. Rasool, and C. Gencel, "A systematic literature review on software measurement programs," *Inf. Softw. Technol.*, vol. 73, pp. 101–121, May 2016.
[30] M. Atalay, U. Murat, B. Oksuz, A. M. Parlaktuna, E. Pisirir, and M. C. Testik, "Digital twins in manufacturing: Systematic literature review for physical–digital layer categorization and future research directions," *Int. J. Comput. Integr. Manuf.*, vol. 35, no. 7, pp. 679–705, Jul. 2022.
[31] E. C. L. Yang, C. Khoo-Lattimore, and C. Arcodia, "A systematic literature review of risk and gender research in tourism," *Tourism Manage.*, vol. 58, pp. 89–100, Feb. 2017.
[32] R. Lovas, A. Farkas, A. C. Marosi, S. Ács, J. Kovács, Á. Szalóki, and B. Kádár, "Orchestrated platform for cyber-physical systems," *Complexity*, vol. 2018, no. 1, Jan. 2018, Art. no. 8281079.
[33] R. Casadei, D. Pianini, A. Placuzzi, M. Viroli, and D. Weyns, "Pulverization in cyber-physical systems: Engineering the self-organizing logic separated from deployment," *Future Internet*, vol. 12, no. 11, p. 203, Nov. 2020.
[34] A. Martín-Gómez, M. J. Ávila-Gutiérrez, and F. Aguayo-González, "Holonic reengineering to foster sustainable cyber-physical systems design in cognitive manufacturing," *Appl. Sci.*, vol. 11, no. 7, p. 2941, Mar. 2021.
[35] V. N. Imrith, P. Ranaweera, R. A. Jugurnauth, and M. Liyanage, "Dynamic orchestration of security services at fog nodes for 5G IoT," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2020, pp. 1–6.
[36] C. Pahl, N. E. Ioini, S. Helmer, and B. Lee, "An architecture pattern for trusted orchestration in IoT edge clouds," in *Proc. 3rd Int. Conf. Fog Mobile Edge Comput. (FMEC)*, Apr. 2018, pp. 63–70.
[37] S. Azimi, C. Pahl, and M. Shirvani, "Particle swarm optimization for performance management in multi-cluster IoT edge architectures," in *Proc. 10th Int. Conf. Cloud Comput. Services Sci.*, 2020, pp. 328–337.
[38] C. Seiber, D. Nowlin, B. Landowski, and M. E. Tolentino, "Tracking hazardous aerial plumes using IoT-enabled drone swarms," in *Proc. IEEE 4th World Forum Internet Things (WF-IoT)*, Feb. 2018, pp. 377–382.
[39] D. Horváth, J. Gazda, E. Slapak, T. Maksymyuk, and M. Dohler, "Evolutionary coverage optimization for a self-organizing UAV-based wireless communication system," *IEEE Access*, vol. 9, pp. 145066–145082, 2021.
[40] J. J. Liang and P. N. Suganthan, "Dynamic multi-swarm particle swarm optimizer," in *Proc. IEEE Swarm Intell. Symp.*, Jun. 2005, pp. 124–129.
[41] J. L. Fernandez-Marquez and J. L. Arcos, "Adapting particle swarm optimization in dynamic and noisy environments," in *Proc. IEEE Congr. Evol. Comput.*, Jul. 2010, pp. 429–434.
[42] E. E. Aydin, O. Kara, F. Cakir, B. S. Cansiz, G. Secinti, and B. Canberk, "Enabling self-organizing TDMA scheduling for aerial swarms," in *Proc. Workshop 8th Workshop Micro Aerial Vehicle Netw., Syst., Appl.*, Jul. 2022, pp. 13–18.
[43] H. Kwong and C. Jacob, "Evolutionary exploration of dynamic swarm behaviour," in *Proc. Congr. Evol. Comput.*, 2003, pp. 367–374.
[44] J. Islam, T. Kumar, I. Kovacevic, and E. Harjula, "Resource-aware dynamic service deployment for local IoT edge computing: Healthcare use case," *IEEE Access*, vol. 9, pp. 115868–115884, 2021.
[45] A. A. Mutlag, M. K. A. Ghani, M. A. Mohammed, A. Lakhan, O. Mohd, K. H. Abdulkareem, and B. Garcia-Zapirain, "Multi-agent systems in fog–cloud computing for critical healthcare task management model (CHTM) used for ECG monitoring," *Sensors*, vol. 21, no. 20, p. 6923, Oct. 2021.
[46] M. Adel Serhani, H. T. El-Kassabi, K. Shuaib, A. N. Navaz, B. Benatallah, and A. Beheshti, "Self-adapting cloud services orchestration for fulfilling intensive sensory data-driven IoT workflows," *Future Gener. Comput. Syst.*, vol. 108, pp. 583–597, Jul. 2020.
[47] C. Aguzzi, L. Gigli, L. Sciullo, A. Trotta, and M. Di Felice, "From cloud to edge: Seamless software migration at the era of the web of things," *IEEE Access*, vol. 8, pp. 228118–228135, 2020.
[48] K. Kuru, "Planning the future of smart cities with swarms of fully autonomous unmanned aerial vehicles using a novel framework," *IEEE Access*, vol. 9, pp. 6571–6595, 2021.

[49] Y. Wang and J. Farooq, "Proactive and resilient UAV orchestration for QoS driven connectivity and coverage of ground users," in *Proc. IEEE Conf. Commun. Netw. Secur. (CNS)*, Oct. 2022, pp. 371–376.
[50] P.-V. Mekikis, P. S. Bouzinis, N. A. Mitsiou, S. A. Tegos, D. Tyrovolas, V. K. Papanikolaou, and G. K. Karagiannidis, "Enabling wireless-powered IoT through incentive-based UAV swarm orchestration," *IEEE Open J. Commun. Soc.*, vol. 4, pp. 2548–2560, 2023.
[51] Z. Liang and Y. Wei, "Research on self-organizing target hunting for mobile robot group," in *Proc. IEEE 4th Int. Conf. Control Sci. Syst. Eng. (ICCSSE)*, Aug. 2018, pp. 67–70.
[52] Q. Qi, J. Liao, J. Wang, Q. Li, and Y. Cao, "Dynamic resource orchestration for multi-task application in heterogeneous mobile cloud computing," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Apr. 2016, pp. 221–226.
[53] D. Ermolenko, C. Kilicheva, A. Muthanna, and A. Khakimov, "Internet of Things services orchestration framework based on kubernetes and edge computing," in *Proc. IEEE Conf. Russian Young Researchers Electr. Electron. Eng. (ElConRus)*, Jan. 2021, pp. 12–17.
[54] Z. Wen, R. Yang, P. Garraghan, T. Lin, J. Xu, and M. Rovatsos, "Fog orchestration for Internet of Things services," *IEEE Internet Comput.*, vol. 21, no. 2, pp. 16–24, Mar. 2017.
[55] S. Taherizadeh, V. Stankovski, and M. Grobelnik, "A capillary computing architecture for dynamic Internet of Things: Orchestration of microservices from edge devices to fog and cloud providers," *Sensors*, vol. 18, no. 9, p. 2938, Sep. 2018.
[56] D. Calvaresi, M. Marinoni, A. Sturm, M. Schumacher, and G. Buttazzo, "The challenge of real-time multi-agent systems for enabling IoT and CPS," in *Proc. Int. Conf. Web Intell.*, Aug. 2017, pp. 356–364.
[57] M. Alam, J. Rufino, J. Ferreira, S. H. Ahmed, N. Shah, and Y. Chen, "Orchestration of microservices for IoT using Docker and edge computing," *IEEE Commun. Mag.*, vol. 56, no. 9, pp. 118–123, Sep. 2018.
[58] U. Arul and S. Prakash, "Toward automatic web service composition based on multilevel workflow orchestration and semantic web service discovery," *Int. J. Bus. Inf. Syst.*, vol. 34, no. 1, pp. 128–156, 2020.
[59] I. Korontanis, K. Tserpes, M. Pateraki, L. Blasi, J. Violos, F. Diego, E. Marin, N. Kourtellis, M. Coppola, E. Carlini, Z. Ledwoń, P. Tarkowski, T. Loven, Y. G. Rozas, M. Kentros, M. Dodis, and P. Dazzi, "Interoperability and orchestration in heterogeneous cloud/edge resources: The ACCORDION vision," in *Proc. 1st Workshop Flexible Resource Appl. Manage. Edge*, Jun. 2020, pp. 9–14.
[60] T. Kiss, P. Kacsuk, J. Kovacs, B. Rakoczi, A. Hajnal, A. Farkas, G. Gesmier, and G. Terstyanszky, "MiCADO—Microservice-based cloud application-level dynamic orchestrator," *Future Gener. Comput. Syst.*, vol. 94, pp. 937–946, May 2019.
[61] A. Cowley and C. J. Taylor, "Orchestrating concurrency in robot swarms," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2007, pp. 945–950.
[62] T. Yeh and S. Yu, "Realizing dynamic resource orchestration on cloud systems in the cloud-to-edge continuum," *J. Parallel Distrib. Comput.*, vol. 160, pp. 100–109, Feb. 2022.
[63] A. Orive, A. Agirre, H.-L. Truong, I. Sarachaga, and M. Marcos, "Quality of service aware orchestration for cloud–edge continuum applications," *Sensors*, vol. 22, no. 5, p. 1755, Feb. 2022.
[64] M. S. de Brito, S. Hoque, T. Magedanz, R. Steinke, A. Willner, D. Nehls, O. Keils, and F. Schreiner, "A service orchestration architecture for fog-enabled infrastructures," in *Proc. 2nd Int. Conf. Fog Mobile Edge Comput. (FMEC)*, May 2017, pp. 127–132.
[65] N. Morkevicius, A. Venč kauskas, N. Satkauskas, and J. Toldinas, "Method for dynamic service orchestration in fog computing," *Electronics*, vol. 10, no. 15, p. 1796, Jul. 2021.
[66] N. Singh, Y. Hamid, S. Juneja, G. Srivastava, G. Dhiman, T. R. Gadekallu, and M. A. Shah, "Load balancing and service discovery using Docker swarm for microservice based big data applications," *J. Cloud Comput.*, vol. 12, no. 1, pp. 1–9, Jan. 2023.
[67] S. Kim, C. Kim, and J. Kim, "Reliable smart energy IoT-cloud service operation with container orchestration," in *Proc. 19th Asia–Pacific Netw. Operations Manage. Symp. (APNOMS)*, Sep. 2017, pp. 378–381.
[68] H. Pan, G. Shi, and J. Ren, "Self-organizing fuzzy control based on modified artificial fish-swarm algorithm," in *Proc. 3rd Int. Conf. Instrum., Meas., Comput., Commun. Control*, 2013, pp. 1562–1567.
[69] A. Haseeb, P. Küngas, and M. Matskin, "Semantic middleware for robot swarm interaction through web services," in *Proc. 27th IASTED Int. Conf. Model. Identificat. Control*, Feb. 2008, pp. 1–6.
[70] A. N. Lam, O. Haugen, and J. Delsing, "Dynamical orchestration and configuration services in industrial IoT systems: An autonomic approach," *IEEE Open J. Ind. Electron. Soc.*, vol. 3, pp. 128–145, 2022.
[71] J. Barbosa and P. Leitão, "Enhancing service-oriented holonic multi-agent systems with self-organization," in *Proc. Int. Conf. Ind. Eng. Syst. Manag.*, 2011, pp. 1–9.
[72] X. Hu and R. Eberhart, "Multiobjective optimization using dynamic neighborhood particle swarm optimization," in *Proc. Congr. Evol. Computation.*, vol. 2, 2002, pp. 1677–1681.
[73] R. Nicholson, T. Ward, D. Baum, X. Tao, D. Conzon, and E. Ferrera, "Dynamic fog computing platform for event-driven deployment and orchestration of distributed Internet of Things applications," in *Proc. 3rd World Conf. Smart Trends Syst. Secur. Sustainablity (WorldS)*, Jul. 2019, pp. 239–246.
[74] W. Du and B. Li, "Multi-strategy ensemble particle swarm optimization for dynamic optimization," *Inf. Sci.*, vol. 178, no. 15, pp. 3096–3109, Aug. 2008.
[75] D. Bratton and J. Kennedy, "Defining a standard for particle swarm optimization," in *Proc. IEEE Swarm Intell. Symp.*, Apr. 2007, pp. 120–127.
[76] C. Li and S. Yang, "Fast multi-swarm optimization for dynamic optimization problems," in *Proc. 4th Int. Conf. Natural Comput.*, 2008, pp. 624–628.
[77] O. M. Bushnaq, A. Chaaban, and T. Y. Al-Naffouri, "The role of UAV-IoT networks in future wildfire detection," *IEEE Internet Things J.*, vol. 8, no. 23, pp. 16984–16999, Dec. 2021.
[78] C. Phan and H. H. T. Liu, "A cooperative UAV/UGV platform for wildfire detection and fighting," in *Proc. 7th Int. Conf. Syst. Simul. Scientific Comput.*, Oct. 2008, pp. 494–498.
[79] A. Khan, "Key characteristics of a container orchestration platform to enable a modern application," *IEEE Cloud Comput.*, vol. 4, no. 5, pp. 42–48, Sep. 2017.
[80] F. Firouzi, B. Farahani, and A. Marinšek, "The convergence and interplay of edge, fog, and cloud in the AI-driven Internet of Things (IoT)," *Inf. Syst.*, vol. 107, Jul. 2022, Art. no. 101840.
[81] M. Driss, D. Hasan, W. Boulila, and J. Ahmad, "Microservices in IoT security: Current solutions, research challenges, and future directions," *Proc. Comput. Sci.*, vol. 192, pp. 2385–2395, Jan. 2021.
[82] I. Čilić, P. Krivić, I. Podnar Žarko, and M. Kušek, "Performance evaluation of container orchestration tools in edge computing environments," *Sensors*, vol. 23, no. 8, p. 4008, Apr. 2023.
[83] B. Anuraj, "Agent-based orchestration on a swarm of edge devices," in *Proc. 17th ACM Int. Conf. Distrib. Event-Based Syst.*, Jun. 2023, pp. 199–202.
[84] F. Hussain, R. Hussain, S. A. Hassan, and E. Hossain, "Machine learning in IoT security: Current solutions and future challenges," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 3, pp. 1686–1721, 3rd Quart., 2020.
[85] A. A. Sadawi, M. S. Hassan, and M. Ndiaye, "A survey on the integration of blockchain with IoT to enhance performance and eliminate challenges," *IEEE Access*, vol. 9, pp. 54478–54497, 2021.
[86] D. Calvaresi, M. Schumacher, and J.-P. Calbimonte, "Personal data privacy semantics in multi-agent systems interactions," in *Advances in Practical Applications of Agents, Multi-Agent Systems, and Trustworthiness. The PAAMS Collection: 18th International Conference, PAAMS 2020, L'Aquila, Italy, October 7–9, 2020, Proceedings 18*. Cham, Switzerland: Springer, 2020, pp. 55–67.

BANANI ANURAJ received the Bachelor of Engineering degree in electrical and electronic from Nanyang Technological University (NTU), Singapore, and the Master of Science degree in electrical engineering from the National University of Singapore (NUS). She is currently pursuing the Ph.D. degree with KU Leuven, Belgium. She has worked in a global setting with employers, such as IBM, KU Leuven, Singapore-MIT Alliance for Research and Technology, NUS, and the Centre for Lifelong Learning and Individualized Cognition, NTU (collaboration with the University of Cambridge). She is a Researcher with the Institute of Informatics, University of Applied Sciences and Arts Western Switzerland (HES-SO), and affiliated with The Sense Innovation and Research Center, Lausanne, Switzerland. Her research interests include e-health, image processing, signal processing, sensor integration, and data analysis. She received the Reputable IEEE Best Paper Award.

DAVIDE CALVARESI received the master's degree in information and automation engineering from Univeristà Politecnica delle Marche, Italy, in 2014, and the Ph.D. degree in emerging digital technologies—real-time embedded systems from the Sant'Anna School of Advanced Studies, Italy, in 2018. He is currently an Associate Professor with the University of Applied Sciences and Arts Western Switzerland (HES-SO). He has been the Primary Investigator of the SEAMLESS Project, aiming at enforcing timing compliance in MAS, and several other (inter)national projects. Currently, he is also the Technical Coordinator of European Project named EXPECTATION, aiming at bridging sub-symbolic and symbolic AI to foster interpretability and explainability in multi-agent systems. Finally, his endeavors in social security pushed him to be the Co-Founder of the startup Wriggle Solutions, providing real-time monitoring of the tires wearing (holding two patents). His research interests include real-time multi-agent systems, explainable artificial intelligence, blockchain, and assistive/rehabilitative technologies. He has been the Chair of the Workshop Real-Time compliant Multi-Agent Systems (RTcMAS2018), the Blockchain for Multi-Agent Systems (BCT4MAS2018–2020), and the EXplainable TRansparent AI and Multi-Agent Systems (EXTRAAMAS2019–2023).

JEAN-MARIE AERTS (Member, IEEE) received the Master of Science degree in bio-engineering and the Ph.D. degree in applied biological engineering from KU Leuven (former Catholic University of Leuven), Belgium, in 2001. He has been a Visiting Researcher with the Engineering Department, Lancaster University, and the Institute of Biomedical Engineering, University of Oxford. Currently, he is heading the Department of Biosystems, KU Leuven, and the Co-Director of the KU Leuven Digital Society Institute. He is also a Co-Promotor of Leuven Health Technology Centre. He is responsible for the master's in human health engineering with the Faculty of Bioscience Engineering, KU Leuven. His research interests include data-based mechanistic modeling of biological systems as a basis for developing controllers and monitors for human health applications.

JEAN-PAUL CALBIMONTE received the Ph.D. degree in artificial intelligence from Universidad Politécnica de Madrid, Spain, in 2013, with a focus on ontology-based data access for sensor data streams. He was a Postdoctoral Researcher with the Distributed Information Systems Laboratory (LSIR), EPFL, from 2013 to 2016. He is currently an Associate Professor with the Institute of Informatics, University of Applied Sciences and Arts Western Switzerland (HES-SO). He is also a Principal Investigator with The Sense Innovation and Research Center, Lausanne, Switzerland. He works on ontology-based approaches for data access and integration on the Web. His research interests include streaming data sources, sensor data on the Web, and the Internet of Things. He has worked extensively on e-health and personalized health solutions based on semantics and intelligent agent-based approaches. He also works on mappings-based integration, heterogeneous data sources, and the generation of linked data, mainly for use cases in eHealth and mHealth.