Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

IoT Capabilities Composition and Decomposition: A Systematic Review

Khalid Halba[1,2], Edward Griffor[3], Ahmed Lbath[1], and Anton Dahbura[2]
[1]Grenoble Alpes University, Grenoble, France (e-mail: khalid.halba@univ-grenoble-alpes.fr, ahmed.lbath@univ-grenoble-alpes.fr)
[2]Johns Hopkins University, Baltimore, Maryland, USA (khalba1@jhu.edu, AntonDahbura@jhu.edu)
[3]National Institute of Standards and Technology, Gaithersburg, Maryland, USA (e-mail: edward.griffor@nist.gov)

Corresponding author: Khalid Halba (e-mail: khalba1@jhu.edu).

ABSTRACT
As billions of IoT devices join the Internet, researchers and innovators increasingly explore IoT capabilities achieved via service composition or reuse of existing capabilities via service decomposition. Many systematic literature reviews (SLRs) were produced on this subject; however, two issues remain to be addressed: i) a reference taxonomy of the different aspects of IoT capabilities composition and decomposition is needed, and ii) many formal questions (e.g., standards role, formal representations applications, state-space explosion countermeasures, etc.), technical questions (e.g., composition process types and automation levels synergies, service decomposition categories, the role of AI/ML, etc.), and QoS questions (e.g., privacy, interoperability, and scalability challenges and solutions, etc.) remain unanswered. We introduce this work by discussing notions of IoT capabilities composition and decomposition in a layered IoT architecture while highlighting the strengths and weaknesses of existing SLRs. We identify unanswered questions through gaps in related work and motivate these questions using the PICOC methodology. We explain the search methodology and organize the topic questions using the proposed reference taxonomy. The identified research questions are answered, and trends and gaps that need additional attention from the research community are highlighted. This effort benefits city planners and end-users of IoT systems as it contributes to a better understanding of the role of composition and decomposition of IoT capabilities in building value-added services or reusing existing ones for resource optimization. For researchers, this effort contributes a reference taxonomy for the topic and sheds light on important questions while highlighting corresponding trends and gaps requiring further attention.

INDEX TERMS Capability, Composition, Decomposition, Internet of Things (IoT), Cyber-Physical Systems (CPS), Systematic Literature Review (SLR).

I. INTRODUCTION

In this section, we highlight the context and background for the IoT capabilities composition and decomposition topic (subsection **A**), define related vocabulary, concepts, and functions while explaining the relationship between these components through a layered architecture and an illustrative usecase (subsection **B**), indicate motivations and key contributions included in this effort (subsection **C**), and provide the timeline and the organization of the SLR study (subsection **D**).

A. CONTEXT AND BACKGROUND

IoT applications bring significant opportunities for different stakeholders (end-users, developers, city planners, etc.) and in numerous smart-city domains (intelligent buildings, smart transportation, smart health, etc.). Innovating new services or reusing existing ones represents two key drivers for different stakeholders, especially developers. Developing novel services requires access to and aggregating basic sensor, service, or device data. IoT capabilities composition aggregates basic data streams from connected devices, sensors, and systems into a composite and value-added service.

An example of IoT capabilities composition is discussed in [224], where authors combined -through a composition function- basic sensor data, including temperature, humidity, air quality, and Wi-Fi signal strength, to innovate a new composite service that they defined as well-being in a smart building.

As for reusability, one service can provide input to different IoT applications or cyber-physical systems: An API for weather conditions can be leveraged and reused (or recycled) across many applications such as hiking recommendation services (when and where to hike), cyber-physical systems such as autonomous vehicles (determine a travel time and plan), employer's notification systems (send alerts to employees about telework options when weather conditions don't facilitate onsite presence). Service decomposition aims to determine basic reusable services in a complex and value-added service.

The processes, functions, and components involved in capabilities composition or decomposition can meet formal (standards, formal specification, and verification, etc.), technical (communication, process type, automation level, etc.), or QoS (scalability, interoperability, privacy, etc.) challenges that we identify and address in this effort through the systematic literature review approach.

B. CONCEPTS AND DEFINITIONS

In this subsection, we shed light on concepts that will be mentioned throughout this effort (IoT, CPS, capabilities classification, composition and decomposition functions), provide a layered architecture that highlights the relationships between the components and functions above, and discuss all these elements through an illustrative use case for clarity.

1) Internet of Things and Cyber-Physical Systems

The **Internet of Things** (**IoT**) [208] is a network of devices connected through the Internet network. This network aims to allow connected devices -IoT devices- to send and receive data in a way that enables sensing and actuation and, as a result, enables innovative services in multiple domains, including health, transportation, and intelligent buildings.

Cyber-physical systems (**CPS**) [38] are engineered systems that integrate physical and digital components with real-time feedback and control loops. CPSs are found in many domains, such as transportation, intelligent buildings, and healthcare. CPSs rely on advanced sensors, actuators, and programs to track and control physical processes, often in sophisticated and dynamic environments. CPSs pose new challenges due to the connectivity component, exposing them to cybersecurity and privacy attacks.

IoT and CPS devices are converging concepts and have multiple synergies in terms of connectivity and compositionality; hence, we might use both expressions interchangeably as explained in [1].

2) IoT or CPS Capabilities

A capability is a feature, service, measurement (temperature, humidity, pressure, etc.), or state of an IoT or a CPS device.

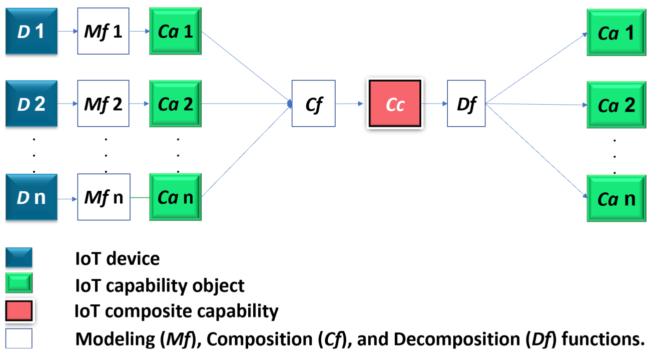

FIGURE 1. An illustration of IoT capabilities modeling, composition, and decomposition functions.

IoT capabilities can also refer to full-fledged complex applications provided by one or multiple devices interconnected at the network layer and orchestrated at the application layer.

3) Capabilities Classification:

Capabilities can be classified into two categories based on how their features can be measured.

i) Tangible capabilities: provide a quantitative metric, such as temperature or round-trip time [12]. Tangible capabilities are usually a characteristic of basic/atomic services such as sensors or network probes. Keeping an open mind when defining what is atomic and what is a composite service is necessary, as certain full-fledged services and applications can be defined as atomic in the context of an even more complex system.

ii) Abstract capabilities: are typically complex services not measured using traditional units. These abstract capabilities are recognized across multiple domains, including smart cities (quality of life [14]), IoT infrastructures (scalability [15]), and smart transportation systems (traffic jam prediction accuracy and driver risk level [16]). Abstract capabilities measurements are often user or programmer-defined in a way that simplifies and makes their assessment intelligible.

4) Composition and Decomposition Functions:

IoT capabilities composition is the art of aggregating existing IoT capabilities to come up with a novel service with added value [19], whereas decomposition aims to reuse a subset of the capabilities of a complex service or distribute its computation [166]. A general diagram is provided in Figure 1, which shows how IoT capabilities are modeled, composed, or decomposed.

To formally explain the different concepts of the topic, let us consider the example of IoT capabilities $Ca1$, $Ca2$, and $Ca3$ and two functions, Cf, and Df, representing IoT capabilities composition and decomposition functions, respectively. $Ca1, Ca2, \ldots, Can$ the atomic capabilities that generate the composite capability Cc. Cc can be considered atomic in the context of a service or an application with a higher level of complexity. In [20], single-function components that are reusable by other city services are packaged and published as standalone components or atomic services, considered as single functional blocks that consume data and implement a feature, such as managing, enriching, or filtering input, and are similar to the concept of a microservice in terms of being a reusable, self-contained piece of software targeting a specific task. In the same effort [20], eight atomic services addressing smart city challenges in data analytics, evaluation, integration, validation, and visualization were pointed out (parking data prediction atomic service, traffic flow predictor; 2 atomic visualization services; 1 data elaboration atomic service; and 3 data transformation atomic services). Functions performed on atomic capabilities include the composition function Cf, and the decomposition functions Df, which can be synchronous [21], asynchronous [22], serial or sequential, probabilistic or alternative, parallel, circular, or cyclic [23] [2]. The success of Cf and Df requires multiple processes, including computations, filtering, ranking, composing, and verifying atomic and composite features [21].

5) A Layered model for IoT Capabilities Composition and Decomposition

Researchers in [24] organized service composition into three layers: physical, virtual object, and service composition. This model represents the composition/decomposition problem because it does not include networking complexities. In [2], the authors based their composition survey on a layered architecture that also considers network and application aspects. In [25], the proposed layered approach, which represents the journey of a capability message, has three levels: i) an information level where the message parameters and temporal scope are defined; ii) a representation protocol level where the message is serialized as a JSON object and made ready for composition by a composition engine; and iii) a session layer where the composite capability is securely delivered to a client or service. A smart home composition framework was proposed in [26] [27], which uses the Majord'home platform as SDN middleware between the data plane layer, where IoT sensors objects reside, and the service composition layer. Other layered models for the composition and decomposition of IoT capabilities were discussed in [16] [15][28].

Based on the efforts mentioned above, we propose in Figure 2 a layered architecture that focuses on composition and decomposition operations and illustrates a hierarchical architecture within which devices, capabilities, applications, and required functions to transition from one layer to another are highlighted: Modeling Mf (turning devices capabilities into composition-ready objects or data models), Composition Cf (aggregating basic capabilities into complex services or applications), and Decomposition Df (reusing/recycling the same atomic capability across multiple complex services).

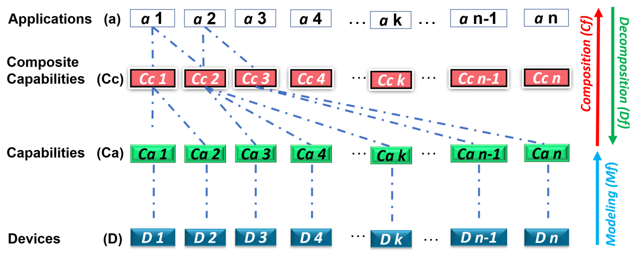

FIGURE 2. Modeling, composition, and decomposition layers and functions.

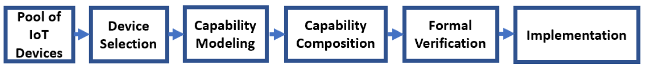

FIGURE 3. Composition of atomic services into composite services.

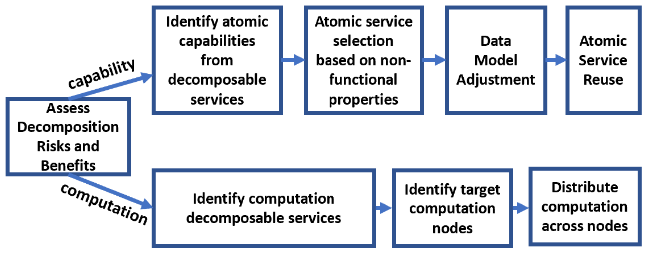
Assess Decomposition Risks and Benefits
capability
Identify atomic capabilities from decomposable services
Atomic service selection based on non-functional properties
Data Model Adjustment
Atomic Service Reuse
computation
Identify computation decomposable services
Identify target computation nodes
Distribute computation across nodes

FIGURE 4. Decomposition of composite services into atomic capabilities.

The bottom layer is the **Devices**, which provides raw and non-composition-ready capabilities. The next layer is the **Capability** model or object layer, where the atomic capabilities are composition-ready using different data models. The third layer is the **Composite Capabilities** layer that incorporates value-added services composed of aggregated capabilities. These composite capabilities are generated using processes or engines that leverage composition-ready capabilities models to aggregate atomic capabilities and provide novel and composite features which can be further composed into more complex entities or **Applications** (e.g., $a1$, $a2$, ..., an in Figure 1), which represent the fourth or upper layer. These applications can be decomposed into less complex or atomic capabilities, or their computation can be distributed across multiple nodes [13]. **Composite capabilities** can also be decomposed into atomic capabilities via decomposition processes.

6) Illustrative use case:

We use the definitions and explanations mentioned above and the formal description in **I.A.4** in the context of a use case to explain the IoT capabilities composition and decomposition topic. Let's consider two applications from Figure 2: $a1$, a mountain hiking recommendation application, and $a2$, a highway travel recommendation application. $a1$ and $a2$ rely on APIs that provide atomic capabilities specific to these applications. For $a1$, $Cc1$ provides oxygen density at a particular altitude. For $a2$, $Cc3$ provides express lane fees based on traffic congestion. $a1$ and $a2$ also rely on APIs that provide input to both applications: $Cc2$ can be a product of the decomposition of a complex service or application such as $a1$ or $a2$. $Cc2$ provides weather information for both applications $a1$ and $a2$ in a way that enables them to output a recommendation on whether or not to hike ($a1$) or whether or not to travel ($a2$).

$Cc1$, $Cc2$, and $Cc3$ are atomic vis-a-vis $a1$ and $a2$ -services with a higher complexity-; on the other hand, $Cc1$, $Cc2$, and $Cc3$ are a composition of multiple elemental/atomic capabilities. For example, $Cc2$ is a composition of $Ca3$ (might provides temperature information), $Ca4$ (might predict precipitation), and Cak (might predict the fog level).

Composition is a bottom-up process that leverages lower-level atomic capabilities to build value-added capabilities that can be composed to create even more complex capabilities or applications. Decomposition is a top-bottom process that analyzes existing composite capabilities components to determine reusable functions or distribute computation.

Dotted lines in Figure 2 trace composition and decomposition paths. Figure 3 and Figure 4 highlight a high-level, step-by-step approach to the composition or decomposition of IoT/CPS capabilities.

C. MOTIVATIONS AND CONTRIBUTIONS

1) Motivations or problems to address

After defining the concepts related to capabilities composition and decomposition, we present in this subsection different motivations or problems that encouraged us to perform this SLR study. While working on new frameworks for capabilities composition in IoT and CPS [193] and studying related use cases [224], we recognized that this topic could be better organized under a taxonomy with three major aspects: Formal, Technical, and QoS. This taxonomy is the first motivation or problem to be addressed for this work as it helps simplify this topic's numerous aspects and sub-aspects (Section IV-A). The second motivation behind this study relies upon pointing out -for each of the aspects above-important questions that were not given enough attention or were not addressed at all; Section III explains in detail the motivation behind answering each SLR question, and Section V discusses these questions, provides corresponding answers, and highlights trends and gaps in the topic.

2) Contributions

The key contributions of this study can be summarized in two points:
i) Providing a reference taxonomy for researchers to better cover, organize, and understand the different aspects and sub-aspects of the IoT capabilities composition and decomposition topic.
ii) Addressing important problems and questions that -to the best of our knowledge- have never been addressed. These problems include uncovering the main reasons for native support for composition and decomposition capabilities by standards and reference architectures; understanding how problems such as state space explosion are solved when modeling complex IoT capabilities; identifying service decomposition benefits to end users and platform builders; understanding the role of AI/ML in improving service decomposition capabilities or processes; and pointing out how interoperability, scalability, or privacy concerns -expressed by end users or other entities- are addressed.

This study provides many benefits for different stakeholders: For IoT platform users, it shows the importance of composition functions in creating value-added services and highlights decomposition functions' role in resource optimization or cost-saving, directly impacting end-users. For programmers and researchers, this study highlights different solutions to hiccups service composition can face at the formal modeling and verification level. Finally, for city planners and associated IoT companies, this study can help solve formal, technical, and QoS problems related to service composition in different smart city domains and in a way that addresses citizens' or customers' needs.

D. TIMELINE AND THE SLR ORGANIZATION

This effort began in April 2018 and will cover the publications on the topic until May 2022. Our interest has grown in service composition while studying composition frameworks [193] and the composition of novel capabilities in the smart building domain [224]. This survey adopts the SLR methodology: Section I introduces the topic of the study; Section II provides related work and highlights the strengths and weaknesses of previous efforts; Section III explains the SLR methodology, including the primary studies selection process and pointing out research questions; Section IV presents the taxonomy of the study, as well as the primary studies, results in a tabulated format; Section V provides answers and a discussion of the SLR questions mentioned in Section III and highlights trends and gaps in the topic as well as threats to this SLR's validity; and Section VI concludes this study by highlighting main achievements while mentioning benefits to different stakeholders and providing a glimpse into our future work.

II. RELATED WORK: STRENGTHS AND WEAKNESSES

In this section, we discuss related work, especially previous SLR studies and Literature reviews that addressed an aspect or more in the topic of IoT capabilities composition and decomposition. Based on the discussion of each SLR and literature review, we revealed components for research questions that received little to no attention in previous efforts (see Table 1). The topics discussed in this SLR can be placed under three aspects: **Formal**, **Technical**, and **QoS**. Some aspects and relative sub-aspects (see Taxonomy in Figure 10) were given substantial attention; therefore, they will not be addressed in this SLR, while other sub-aspects were scarcely or not surveyed at all: we picked some of the less addressed or non-addressed topics and crafted corresponding SLR questions.

A. FORMAL GAPS

Regarding the **Formal** aspect, many previous works tackled corresponding sub-aspects, including IoT composition standards and frameworks [4][213][9], composition algorithms (heuristic, meta-heuristic, exact, hybrid) [178], formal verification tools, techniques, and properties verification such as correctness [5][10] or security [7]. None of the SLR surveys addressed the motivation behind native support for composition or decomposition guidelines and mechanisms by standards, frameworks, and architectures (RQ1). In addition, previous SLR questions did not comprehensively explain the main properties of formal representations in service composition from a modeling and formal verification perspective (RQ2). Discussing recent formal trends in service composition, including the properties type, formal modeling approaches, formal verification tools, and implementations, requires an up-to-date revision (RQ3), and the question of state space explosion and how it was tackled and to what extent it was solved was not discussed in previous efforts, which this SLR study attempts to address (RQ4). We put service composition sub-aspects that relate to standards, frameworks, architectures, and formal verification techniques and challenges under the Formal aspect bucket.

B. TECHNICAL GAPS

For the **Technical** aspect, the sub-aspects previously discussed include service composition in the cloud [6] and industrial environments [196], composition service types, attributes, domains of application [3], composition planning and strategies [11] [195] [228], composition platforms [213], and composition models, techniques, and tools [177][217]. However, none of these efforts have addressed some Technical sub-aspects, including stakeholders' concerns regarding service composition (RQ5). Similarly, no SLR question comprehensively discussed the nature of composition platforms and how composition implementations differ within these different categories of platforms (RQ6). In addition, the relationship between composition automation levels and composition process types was not highlighted in previous SLRs (RQ7). In addition, the role of communication protocols in composing services at different layers of IoT environments needs to be discussed (RQ8). A comparative study of the different roles data models perform in service composition is also lacking (RQ9). From a measurability perspective, composite services typically reflect a metric that is difficult to assess using conventional metrics, and this aspect is also missing in the studied SLRs (RQ10). Similarly, the decomposition of services for reuse or resource optimization has never been surveyed (RQ11). Finally, the role of artificial intelligence and machine learning (AI/ML) in capabilities composition, either in the composition process or the nature of capabilities themselves, was not surveyed. To the best of our knowledge, this is the only manuscript that address it (RQ12).

C. QOS GAPS

Regarding the **QoS** aspect, different SLRs addressed key QoS questions under different themes, including functional and non-functional properties. For example, the availability of composite services was studied in surveys [5] [11] [196]. The cost was studied in [5] [11] [196] [6], time-related QoS questions, including execution time, response time, and latency, were addressed in [5] [6] [11] [196], reliability and reputation were both discussed in [11] [196], and scalability was addressed in [6],[195],[196]. Unique QoS properties that were addressed in SLR questions in previous surveys include performance parameters such as completeness, distribution, dynamicity, level of automation, maturity, QoS awareness [195], efficiency, and optimization [6], and security and throughput [11]. By looking at what was covered in previous SLR efforts as well as the existing literature, we identified QoS questions that were not addressed previously; this includes scalability challenges in terms of service composition (RQ13). Although the authors in [2][195] discussed which composition efforts provided high, low, or no scalability, they did not address the challenges that face composition approaches to satisfy increased levels of scalability (in terms of latency, computation, etc.). Similarly, in [5] and [196], ensuring a high level of scalability while maintaining a low response time and low verification cost are examples of scalability challenges for service composition that need to be addressed. Another significant aspect that was not given a comprehensive assessment was the interoperability challenges and solutions (RQ14). Different efforts addressed specific areas of the interoperability question, which is the case with SLR [3], which mentioned some interoperability challenges, including differences in network protocols, data models, and service types. SLR [3] also defines a fully interoperable composition as "service type heterogeneity". Similarly, SLR [2] suggested open-source frameworks and a dynamic service composition ensuring interoperability. SLR [5] addressed formal verification challenges related to the interoperability of to-be-composed IoT capabilities. SLR [228] mentioned the integration, selection, and discovery of services as challenges to interoperability. By answering (RQ14), we provide a consolidated response to interoperability challenges and solutions. Finally, an aspect of crucial importance in the age of data sharing is the privacy challenges and solutions when composing services. Although many studies have addressed this concern from a composition perspective [231] [232], none of the SLR surveys addressed privacy-related service composition questions. We address this aspect in RQ15, try to understand how different research efforts improve privacy in terms of service composition and explain how new technologies such as blockchain can be leveraged to improve privacy.

D. RELATED WORK: SUMMARY

Table 1 aggregates survey efforts that tackled IoT and CPS capabilities composition based on the survey type, the year

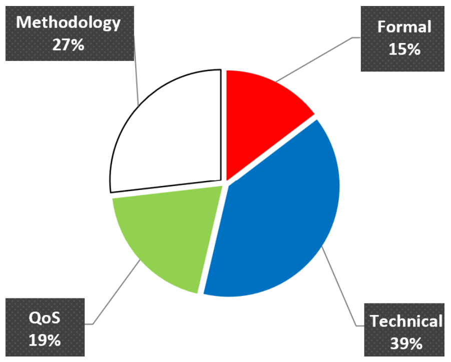

FIGURE 5. Distribution of previous SLRs questions per the proposed taxonomy aspects and the SLR methodology

when the research was conducted, the topics covered in the survey, the covered period of the study, and strengths as well as gaps in each study that inspired the SLR questions in this work. The takeaways concluded from Table 1 include: i) a strong interest in answering specific questions related to the topic (SLR surveys) compared to simply summarizing aspects related to it (Literature reviews), ii) there is a strong interest in the Technical aspect of the topic compared to Formal and QoS aspects, and iii) the topic of IoT capabilities composition and decomposition is still trending as it was continuously discussed as early as 1992 until this year (2022) and still attracts the interest and curiosity of researchers. To complete our related work analysis, SLR efforts investigated in Table 1 proposed a total of 41 SLR questions. None of the questions in our survey have been addressed previously. Figure 5 shows the distribution of the previous SLR questions based on the aspect under which they fall. The questions related to the methodology were geared toward the SLR methodology itself or were too general to organize under a specific aspect.

III. RESEARCH METHODOLOGY

The SLR approach uses an objective research methodology to answer specific research questions based on relevant papers on that topic. SLR reviews require expertise in the domain of study, search in different databases, and require years to produce. However, literature reviews can use subjective research methods to summarize topics using informal approaches. The SLR approach was deemed the most suitable for conducting this survey, as the main goal is not to summarize aspects of the studied topic but to address specific unanswered questions related to formal, technical, and QoS sub-aspects of IoT capabilities composition and decomposition. The guidelines proposed by Kitchenham [226] [229] [230] were used, as well as guidelines from the SLR studies in the related work for respecting the SLR methodology: A) formulating the research questions based on the PICOC approach [226][227][234], and B) explaining the search process while highlighting inclusion and exclusion criteria, C) performing Quality assessment, D) discussing the effort limitations, E) collecting data, F) analyzing data, and G) executing the SLR approach.

A. FORMULATING THE RESEARCH QUESTIONS

Based on the gaps and weaknesses of related work, formal (RQ1-RQ4), technical(RQ5-RQ12), and QoS(RQ13-R15) questions -that were not addressed in previous SLR efforts- were pointed out, and the list of these questions was elaborated in Table 2 along with corresponding motivations.

B. EXPLAINING THE SEARCH PROCESS AND THE INCLUSION/EXCLUSION CRITERIA

1) Explaining the search process

The research questions (RQs) in Table 2 and the corresponding taxonomy aspects and sub-aspects presented in Figure 10 are the foundations of this SLR review because they guide the search process by guaranteeing that the selection of primary studies is directly related to the SLR questions. The search process was performed in 6 stages:

• **a)** In Stage 1, the SLR questions and the corresponding taxonomy aspects and sub-aspects are identified.

• **b)** In Stage 2, the search databases are selected, and the corresponding search string and filtering formula are composed as illustrated in Figure 6. The search string was constructed as follows:

→ We started by crafting the **main search sentence** of the topic : **Service Composition or Decomposition in IoT.** Titles, Abstracts, or Keywords must contain the words (or corresponding synonyms) that compose the main search sentence.

→ For every word in the **main search sentence**, we identified a number of synonyms:

- Service: Capability, Feature, Function.

- IoT: Internet of Things, Cyber-Physical Systems, CPS.

- Composition : Aggregation.

- Decomposition: We couldn't find relevant synonyms to this word, and we ended up using it exclusively to prevent false positives.

→ Plural forms of certain words are considered. For example, in the search string, the words **Capability** or Capabilities can be searched through the expression: **Capabilit*.**

→ For each SLR question, the primary studies filtering formula can be expressed as follows: **main search sentence** + **Population keywords** (Check Table 3, 3rd Column).

→ For each database (SCOPUS and Google Scholar), we created corresponding search strings (check Figure 6). It's worth mentioning that document type, language, and subject area limitations were automatically added to the SCOPUS **main search sentence'** search string before filtering on **Population keywords** from Table 3.

• **c)** In Stage 3, we executed the SCOPUS search string for the **main search sentence**, which focused on the title, abstract, and manuscript keywords as explained in Stage 2, this yielded 2805 manuscripts.

• **d)** In Stage 4, the search results from Stage 3 are narrowed by applying different filtering keywords in column **Population** from Table 3. For example, QoS/privacy-related primary studies were extracted by adding different related

TABLE 1. Related work: Literature and SLR studies that tackled the IoT/CPS capabilities composition topic.

Ref	Type	Year	Topic	Period	Strengths (✓) and Gaps (x)
This effort	SLR	2022	Formal, Technical, and QoS aspects of IoT Capabilities Composition and Decomposition	2006-2022	(✓) A new reference taxonomy based on Formal, Technical, and QoS aspects. (✓) Based on gaps in existing research, 15 formal, technical, and QoS questions were identified and answered. (Check Table 2 for motivations behind each SLR question). **(x)** Check gaps in V.B.2).
[2]	SLR	2018	Functional and non-functional service composition properties	1993-2018	(✓) Multiple formal, technical, and QoS sub-aspects addressed. **(x)** QoS: privacy was considered a challenge and was left for future work.
[3]	SLR	2019	Service composition in interoperable and heterogeneous environments.	1992-2019	(✓) Strong focus on service composition interoperability challenges in specific service types (REST, SOAP, EOS). **(x)** The SLR question does not answer how data and protocol interoperability challenges in service composition have been addressed or resolved in previous efforts.
[4]	Literature Review	2019	Composition types, models, standards, and QoS sub-aspects.	1996-2018	(✓) A comparative study of service composition approaches. **(x)** The SLR approach is missing because it was a literature review.
[5]	SLR	2018	Formal verification role in assessing service composition correctness.	1999-2018	(✓) Formal sub-aspects in service composition addressed. **(x)** An SLR question to address the state space explosion problem in service composition is missing.
[6]	SLR	2017	Cloud services composition	2003-2017	(✓) Focus on cloud composition technologies. **(x)** Compositions at the Edge, Fog, SDN, and simulation platforms were not addressed.
[7]	Literature Review	2020	Formal verification of IoT protocols	1976-2020	(✓) Focus on verifying the security of compositions and tools leveraged for this end. **(x)** The SLR approach is missing because it was a literature review.
[8]	Literature Review	2017	Interoperability approaches in the IoT application layer	2009-2016	(✓) Application layer composition standards and frameworks compared **(x)** The SLR approach is missing because it was a literature review.
[9]	Literature Review	2020	SOA Capabilities composition and formal specifications.	2003-2020	(✓) Explaining the differences between SOA service composition languages. **(x)** An architecture agnostic comparison cloud has been more comprehensive (SOA, REST, ..). **(x)** The SLR approach is missing because it was a literature review.
[10]	Literature Review	2019	Formal verification approaches for composed IoT services.	2015-2019	(✓) IoT composite services correctness verification was the focus of the study. **(x)** The SLR approach is missing because it was a literature review.
[177]	Literature Review	2021	Comparison of enterprise service composition models in IoT	1992-2020	(✓) Composition techniques, models, and tools were highlighted and compared. **(x)** Formal sub-aspects and measurability/assessment of QoS metrics are not addressed.
[11]	SLR	2015	QoS-aware web service composition	2005-2015	(✓) A comprehensive study of QoS-Aware service composition algorithms (heuristic, meta-heuristic, etc.)
[195]	SLR	2022	Web service composition	1994-2021	(✓) Focus on Technical sub-aspects of web-service composition. **(x)** Formal sub-aspects are missing.
[196]	SLR	2022	Service composition methods in cloud manufacturing systems	2008-2021	(✓) Focus on Technical sub-aspects in composing cloud manufacturing capabilities. **(x)** Formal and measurability aspects are missing.
[213]	Literature Review	2014	Web services composition	1997-2014	(✓) A summary of standards, prototypes, and web-service composition platforms. **(x)** Motivations for native support of composition by standards is missing **(x)** The SLR approach is missing because it was a literature review.
[217]	Literature Review	2015	Web services composition tools and techniques	1974-2015	(✓) Overview of service composition techniques, technologies, and tools. **(x)** SLR approach missing since it is a literature review.
[228]	SLR	2021	QoS-Aware Service Composition	2008-2020	(✓) Focus on the technical and QoS aspects related to Hybrid meta-heuristic composition algorithms in SOA. **(x)** Formal sub-aspects not covered.

keywords (privacy, private, etc.) to the search string of the **main search sentence**. Performing filtering on the different aspects and sub-aspects resulted in 553 manuscripts. Some papers contained keywords related to the research questions, but only 50 primary studies were kept as they substantially address one or more RQ in a specific paragraph or as the main topic of the primary study [257].

• **e)** In Stage 5, more relevant manuscripts were included using the forward and backward snowballing techniques based on the 50 primary studies in Stage 4 to find more answers to the research questions, which required reading the full text of these publications instead of focusing on the abstract, title, or keywords, which added 103 more manuscripts.

• **f)** In Stage 6, and for completion, 29 additional manuscripts were included using the search script crafted for the Google Scholar database.

TABLE 2. Systematic literature review (SLR) questions (RQs) and corresponding motivations.

Research Questions	Motivations
RQ1: What is the motivation for native support for IoT capabilities composition/decomposition mechanisms by standards, reference models/architectures (RMAs), and frameworks?	• Service composition/decomposition platforms rely on different frameworks, standards, or reference architectures that may -or not- support composition mechanisms and guidelines. • Answering this question would encourage future standard groups to consider composition/decomposition benefits during the research and writing phase and platform builders to adopt standards or frameworks that keep composition and decomposition guidelines in mind during the implementation.
RQ2: What are the main properties of formal representations leveraged in service composition?	• Explaining the role formal representations play in modeling and explaining composite capabilities from functional and non-functional perspectives, and highlighting how formal representations can be leveraged in verifying composite capabilities' formal properties such as correctness.
RQ3: What are the current trends in formal representations modeling and formal verification?	• As there is a wide range of techniques and tools adopted for enabling formal specification and studying different properties, the goal is to make researchers and engineers recognize these elements for potential use in their research efforts or industrial applications.
RQ4: What are the different ways and how effective formal verification techniques and tools tackled the state-space explosion problem in service composition?	• What motivates this question is informing researchers of the different solutions used to solve or minimize the impact of the state space explosion on service composition verification, as the number of states explodes with the complexity that comes with composing different atomic capabilities with a wide range of values and states.
RQ5: What are the different stakeholders' categories and concerns when it comes to composing or consuming capabilities in different domains?	• Different stakeholders deal with service composition challenges from their own perspectives. Developers, Users, City Planners, and Researchers each have their own concerns and expectations. Understanding these concerns from the get-go would help in addressing them and taking them into consideration either while developing new platforms for composite services or when using these solutions by end users.
RQ6: What are the technical differences in capabilities composition implementation in different platforms?	• Composing capabilities in the cloud differs from composing capabilities in the edge or the fog. Responding to this question would enlighten researchers and engineers on which processes or services should be implemented in which composition layer.
RQ7: What are the different composition process types, and how do they differ in terms of automation level?	• Service composition can be synchronous or asynchronous, rule-based, or programming-based, among other process types. These process types are discussed in light of the automation level. Based on existing literature, we check whether automating composition can be better performed under a particular composition process type.
RQ8: What roles do communication protocols play in composing or decomposing IoT capabilities?	• Besides ensuring communication between different components involved in service composition, the other roles communication protocols play -either in improving certain QoS properties or enabling some other capabilities- are discussed.
RQ9: What are the roles of data models leveraged in service composition and decomposition?	• IoT data models differ in terms of expressiveness and complexity, among other properties. Leveraged data models in service composition are highlighted as well as their roles in the context in which they were leveraged. This will help developers make informed choices relative to capabilities data models when building new capabilities in the IoT space.
RQ10: How are atomic or composite capabilities quantified or measured?	• Composite capabilities typically lack conventional methods of measurement or assessment compared with atomic capabilities. Answering this question would inspire and inform researchers about different ways of assessing and measuring the performance or levels of composite capabilities.
RQ11: What are the benefits of building IoT platforms and complex services with decomposition in mind?	• Service composition has been extensively discussed in previous surveys and literature. However, to the best of our knowledge, this is the only study that extensively addresses service decomposition. We explain decomposition flavors and the benefits it brings (including reuse) when services are built with it in mind.
RQ12: What role can AI/ML techniques play in shaping or improving service composition?	• To the best of our knowledge, this is the only survey to address the role of AI/ML in service composition. Two ways in which AI/ML plays a role were identified: improving service composition workflow components (e.g., service selection) or building services with AI/ML capabilities.
RQ13: What are the main scalability challenges and solutions adopted when composing IoT and CPS capabilities?	• Different Technical sub-aspects can either improve or hinder scalability when composing services. Based on existing efforts, these challenges and solutions are revealed to help composite capabilities stakeholders build and use features that scale.
RQ14: What are interoperability challenges and solutions when composing capabilities from heterogeneous environments?	• Composing capabilities requires interoperable data models, network APIs, and synchronized data, among other requirements. Those requirements are exposed to inform IoT platforms builders of interoperability considerations when it comes to composing novel IoT capabilities.
RQ15: What are the main privacy challenges and solutions in service composition?	• Privacy is an end-user concern that is receiving increasing attention, especially with the advent of new standards such as GDPR that impact how IoT systems should be built to address privacy [233]. We identify privacy concerns in service composition as well as the role of new technologies or best practices, such as the blockchain or regulations, in addressing these concerns.

Figure 9 highlights the search stages: the 182 primary studies are highlighted in the **Ref** column from Table 4 to 18, with some primary studies providing answers to more than one RQ. The primary studies for each RQ are highlighted in Table 3 column **Primary Studies**. For more details about the primary studies examined in this effort, readers can refer to [257], where we put together the list of primary studies, as well as related information (citation ID in this manuscript, year of publication, the source database, how each study was extracted (Main Search, Snowballing, Manual), publisher, as well as its role in answering a Research Question (RQ) in this SLR.). For the other referenced material in this manuscript, publications and links mentioned when introducing certain

TABLE 3. Criteria, Primary Studies, and scope of the SLR: **P**opulation, **I**ntervention, **C**omparison, **O**utcome, **C**ontext (**PICOC**).

Criteria	Primary Studies	Population	Intervention	Comparison	Outcome	Context
RQ1	Check Table 4 Column **2**	Formal: Frameworks, Architectures, Standards	Classifying and characterizing IoT service composition and decomposition efforts based on a taxonomy, and answering key questions related to the taxonomy aspects based on relevant literature by extracting and discussing data.	A comparison study connecting primary studies in the IoT/CPS service composition and decomposition topic to come up with answers to important formal, technical, and QoS questions	Classifying, comparing, and analyzing data from different research efforts to answer the key questions identified. Trends, gaps, and Future work will also be discussed.	The systematic literature review approach is leveraged to investigate and consolidate resulting primary studies in IoT or CPS service composition within the context of 3 aspects : Formal, Technical, and QoS.
RQ2	Check Table 5 Column **2** and Table 6 Column **1**	Formal : Formal Representations				
RQ3	Check Table 5 Column **2** and Table 6 Column **1**	Formal : Formal Verification Techniques, aspects, applications, tools				
RQ4	Check Table 7 Column **1**	Formal : State Space Explosion				
RQ5	Check Table 8 Column **2**	Technical : Stakeholders				
RQ6	Check Table 9 Column **2**	Technical :Composition Platforms				
RQ7	Check Table 10 Column **2**	Technical : Composition Processes and Automation				
RQ8	Check Table 11 Column **3**	Technical : Communication Protocols				
RQ9	Check Table 12 Column **2**	Technical : Data Models				
RQ10	Check Table 13 Column **1**	Technical : Measurability				
RQ11	Check Table 14 Column **1**	Technical : Decomposition				
RQ12	Check Table 15 Column **1**	Technical : AI/ML				
RQ13	Check Table 16 Column **1**	QoS : Scalability				
RQ14	Check Table 17 Column **1**	QoS : Interoperability				
RQ15	Check Table 18 Column **1**	QoS : Privacy				

concepts, web sources, and Git repositories are not counted as primary studies, but they are components for explanation and completion.

2) Inclusion Criteria

From a content perspective, the inclusion criteria require:
• Relevance to the 15 formal, technical, and QoS research questions or the taxonomy sub-aspects.
• The manuscript addresses an RQ or taxonomy sub-aspect as the main component or at least in a specific section/paragraph.
• The manuscript exists in the SCOPUS or Google Scholar Database.

3) Exclusion Criteria

• SCOPUS main search: non-peer-reviewed publications and document type limitations: sources that are not Conference Papers (cp), Articles (ar), Book Chapters (ch), or Books (bk).
• Google Scholar Manual Search: respected the same criteria as in the SCOPUS main search while tolerating a few important technical reports.
• All databases: document Type limitations: MS or PhD dissertations, white papers, SLR and Literature reviews, documents that are not in the field of Computer Science, Engineering, or Mathematics, and manuscripts written in languages other than English.
• All databases: the full text of the candidate primary study does not provide sufficient information to allow classification of the studied sub-aspect properties.
• All databases: for manuscripts selected manually or using snowballing, the full text of the candidate primary study could not be obtained by contacting the authors or other means.

(A) : SCOPUS Search String

```
TITLE-ABS-KEY ( ( ( service* ) OR ( capabilit* ) OR (
feature* ) OR ( function* ) ) AND ( ( composition* ) OR (
aggregation* ) OR ( decomposition* ) ) AND ( ( iot ) OR (
internet AND of AND things ) OR ( cps ) OR ( cyber AND
physical AND system* ) ) ) AND ( LIMIT-TO ( DOCTYPE , "ar" )
OR LIMIT-TO ( DOCTYPE , "cp" ) OR LIMIT-TO ( DOCTYPE , "ch" )
OR LIMIT-TO ( DOCTYPE , "bk" ) ) AND ( LIMIT-TO ( SUBJAREA ,
"COMP" ) OR LIMIT-TO ( SUBJAREA , "ENGI" ) ) AND ( LIMIT-TO (
LANGUAGE , "English" ) )
```

(B) : Google Scholar Search String

```
( ( ( service* ) OR ( capabilit* ) OR ( feature* ) OR (
function* ) ) AND ( ( composition* ) OR ( aggregation* ) OR (
decomposition* ) ) AND ( ( iot ) OR ( internet AND of AND
things ) OR ( cps ) OR ( cyber AND physical AND system* ) ) )
```

(C) : Filtering sub-aspects or RQs

```
(Search String ((A) OR (B)) (AND) (RQ Sub aspect Keywords)
```

FIGURE 6. Search Strings and filtering method.

C. QUALITY ASSESSMENT

The six steps followed in the research process were intended to ensure that only relevant and high-quality manuscripts were selected as primary studies. Techniques to ensure quality include full-text reading, forward and backward snowballing, and manual selection of relevant papers that add value to the RQs answers. We were inspired by the quality assessment elements proposed in IEEE Access SLR [228] to ensure that the selected articles respect a minimum quality threshold of 75% of the criteria below :

i) Validating the data source: Queried databases and journals are well-known and trusted by the research community through indicators such as the impact factor.

ii) Relevance to the research domain (IoT and Cyber-physical Systems).

iii) Presence of substantial information: The sole presence

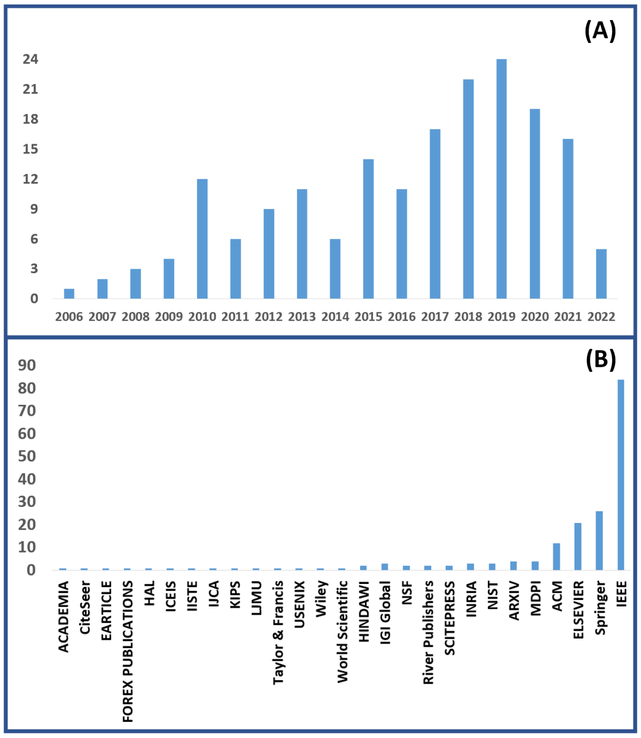

FIGURE 7. Count of primary studies per year and per publisher.

of RQ keywords doesn't imply inclusion.

iv) Primary studies selected provide solid contributions that address the SLR objectives.

To ensure that the SLR is inclusive, efforts that do not originate from well-known -but genuine and trusted- publishers were included; only 16% of the primary studies were obtained using a manual search on the Google Scholar database to ensure that the majority of primary studies were systematically selected while guaranteeing a level of quality and completeness by including manually selected - and RQ relevant- manuscripts. Although important, the number of citations was not taken into consideration as an exclusion criterion as that would discriminate against high-quality or newer publications that might have important data; the same reasoning applies to the year of publication of the manuscripts, which could limit the scope of the study with no concrete benefit. The resulting primary studies were published between 2006 and 2022, with more than 70% being published between 2015 and 2022, and for each SLR question, the majority of primary studies that address them span this period which would reveal the latest advances in the topic and provide up-to-date answers to the identified SLR questions.

Figure 7 (A) shows the number of publications per year; this distribution shows that more than 70% of scientific publications on this topic were published after 2015; hence, the continued relevance of the topic in recent years.

Figure 7 (B) shows the count of primary studies publications per publisher, primary studies from less-known publishers -representing 21% of primary studies- to achieve a higher level of completeness and to account for the importance of these studies, while 79% of the manuscripts were extracted from well-known publishers (ACM, ELSEVIER, IEEE, and SPRINGER) to guarantee a high level of quality.

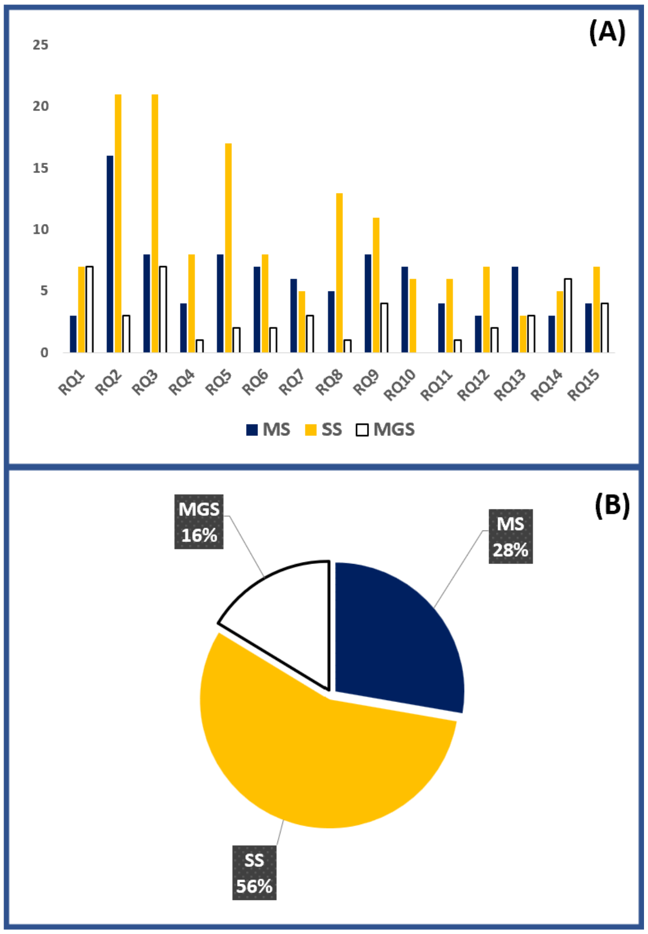

FIGURE 8. Distribution of primary studies per search method: Main SCOPUS (MS), Snowballing SCOPUS (SS), Manual Google Scholar (MGS).

Figure 8 (A) accounts for primary studies cited in more than one RQ and shows the distribution of primary studies search methodology per RQ: Main (MS), Snowballing (SS), Manual (MGS).

Figure 8 (B) highlights the percentage of the primary studies search methodology: 84% of the primary studies were either obtained using the direct search string and filtering keywords or the snowballing technique, and all the results of the main search or the snowballing search are indexed in the SCOPUS database.

D. LIMITATIONS

The limitations of this SLR are as follows:

- Not including manuscripts ruled out by exclusion criteria.
- Some formal, technical, and QoS sub-aspects (referred to

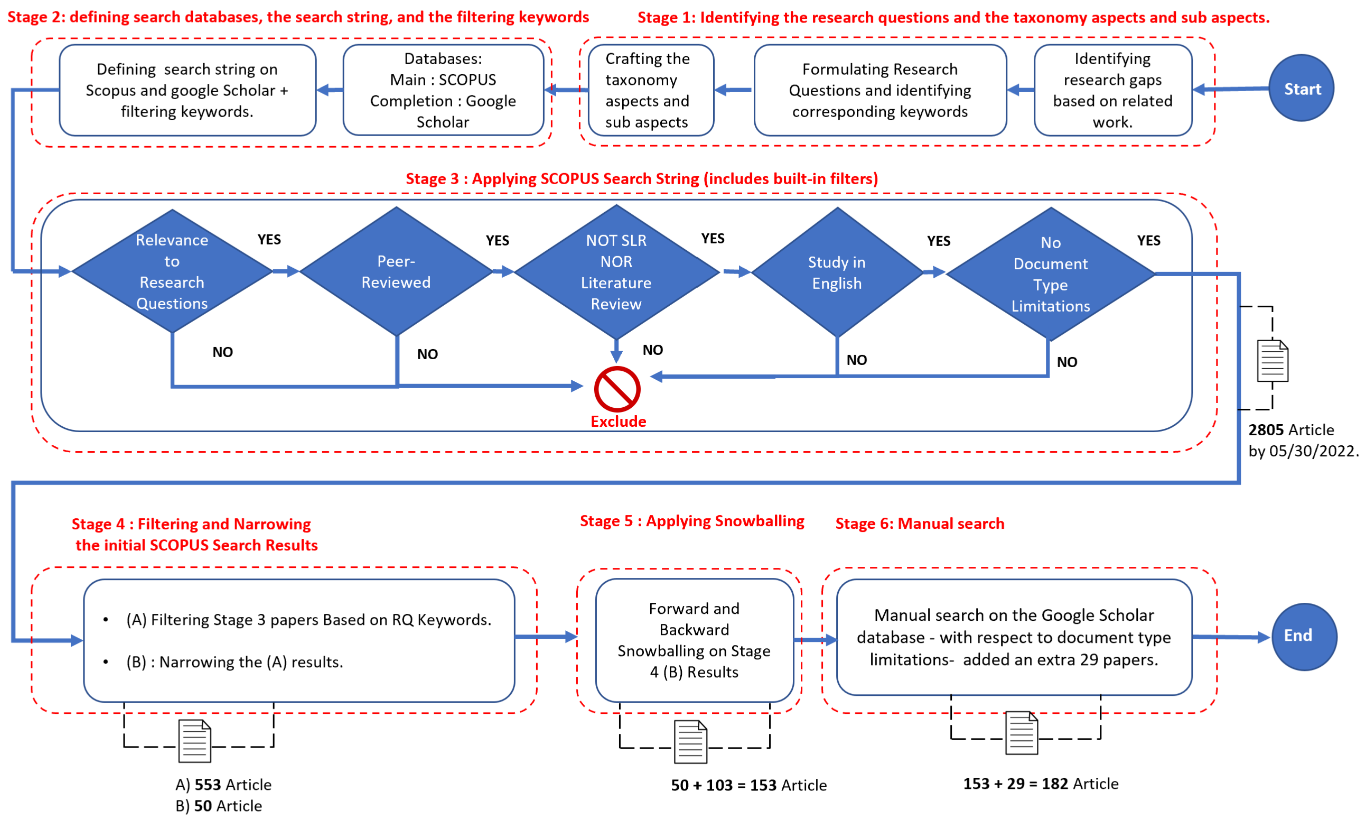
Stage 2: defining search databases, the search string, and the filtering keywords
Stage 1: Identifying the research questions and the taxonomy aspects and sub aspects.
Start
Identifying research gaps based on related work.
Formulating Research Questions and identifying corresponding keywords
Crafting the taxonomy aspects and sub aspects
Databases: Main : SCOPUS Completion : Google Scholar
Defining search string on Scopus and google Scholar + filtering keywords.
Stage 3 : Applying SCOPUS Search String (includes built-in filters)
Relevance to Research Questions
Peer-Reviewed
NOT SLR NOR Literature Review
Study in English
No Document Type Limitations
YES
NO
Exclude
2805 Article by 05/30/2022.
Stage 4 : Filtering and Narrowing the initial SCOPUS Search Results
(A) Filtering Stage 3 papers Based on RQ Keywords.
(B) : Narrowing the (A) results.
A) 553 Article
B) 50 Article
Stage 5 : Applying Snowballing
Forward and Backward Snowballing on Stage 4 (B) Results
50 + 103 = 153 Article
Stage 6: Manual search
Manual search on the Google Scholar database - with respect to document type limitations- added an extra 29 papers.
153 + 29 = 182 Article
End

FIGURE 9. Multi-stage selection process with the inclusion/exclusion criteria.

in the taxonomy as Other Formal, Other Technical, and Other QoS) are not discussed (out of scope)
• Databases queried (Only SCOPUS and Google Scholar).
• Some papers were probably not considered due to human error while generating results using the search strings or while selecting papers.
• Although we strongly believe that we extensively covered the studied sub-aspects, relevant papers after May 2022 might have been missed due to the consolidation phase.

E. DATA EXTRACTION

Data in the Results section were extracted from 182 primary studies; the methods used included filtering on the SCOPUS and Google Scholar databases. Keywords leveraged for extraction include those related to research questions but also : (i) title, (ii) names of authors, (iii) year of publication, (iv) Publication venue and related quality index (v), and approaches, criteria, and parameters for each sub-aspect or research question.
Extracted data were placed in tables by referencing the primary studies, as well as other columns, to classify and compare the different studies based on each RQ requirement. For accuracy purposes, the extracted data were reviewed by the main author and agreed upon by the co-authors.

F. ANALYZING DATA

Answering the research questions of this SLR required analyzing tabulated data resulting from data extraction and synthesizing their content based on the RQ requirements and response elements. The main output of the analysis is exposing techniques, constraints, solutions, and other aspects and properties that provide elements for answering each RQ. All documents were subject to classifications in the tabulated data for each RQ, and this classification was re-evaluated by all authors for refinement.

G. EXECUTION

This SLR was conducted in five incremental updates; the first execution was done in April 2018 (yielded 61% of the 182 primary studies), the second in February 2020, the third in June 2021, and the 4th in December 2021 after the reviewers' feedback, and the last update was performed on May 30th, 2022, with a full reevaluation of the abstracts. After the last update, a re-evaluation of the primary studies identified a total of 11 false exclusions, which were later included in the final result of 182 selected documents.

IV. RESULTS: TAXONOMY AND SURVEYED ASPECTS DATA

In this section, we **A)** propose a taxonomy for the IoT capabilities composition and decomposition topic based on the studied RQs, as well as an overview of references cited in this effort, including primary studies. Then, in subsection **B**, we provide extracted data for the Formal sub-aspects, which answers formal questions RQ1-RQ4. In subsection **C**, the extracted data that would answer Technical questions RQ5-RQ12 is provided, and finally, in subsection **D**, we provide tabulated data for the QoS sub-aspects that would help answer RQ questions RQ13-RQ15.

A. A TAXONOMY OF THE SLR RESEARCH QUESTIONS ASPECTS AND SUB-ASPECTS AND EXTRACTED DATA DISTRIBUTION.

1) Taxonomy

Based on related work and our experts' opinions, the issues and research questions addressed in this survey are organized based on three aspects: Formal (RQ1-RQ4), Technical (RQ5-RQ12), and QoS (RQ13-RQ15). We were inspired by previous SLRs on how they organized topics into taxonomies [3] [4] [6] [7] [8] [11] [195] [196] [213], and we proposed in Figure 10 a taxonomy for the IoT capabilities composition and decomposition topic (root), with an indication of which sub-aspect relates to which research question. The taxonomy would help in the search/filtering steps and guide the discussion and trend analysis. We believe that the proposed taxonomy can be extended by researchers to become comprehensive (with the inclusion of the other formal, technical, and QoS sub-aspects not discussed in this SLR) and can be leveraged by researchers to build a full picture of service composition formal, technical, and QoS sub-aspects. The taxonomy's three aspects (leaves) and corresponding sub-aspects (sub-leaves) discussed in this SLR are as follows:

•**Formal Aspect**: We include service composition-related standards, frameworks, and reference architectures as well as formal verification -which includes formal specification languages, formal verification techniques, and challenges-under one aspect as they all aim to provide knowledge and common ground for representing or building a certain concept (composition algorithms fall under this aspect but are not addressed in this effort). The Formal aspect of the taxonomy contains the following RQs-related / sub-aspects: building composite capabilities based on a certain standard, framework, or reference architecture (RQ1); highlighting formal representations properties (RQ2); identifying trends related to formal verification of capabilities composition or decomposition (RQ3), and studying formal verification constraints (the state space explosion problem) (RQ4).
•**Technical Aspect**: represents sub-aspects including service composition domains and stakeholders (RQ5), service composition platform nature (RQ6), service composition automation and process type (RQ7), communication protocols leveraged in service composition (RQ8), capabilities data models (RQ9) and measurability aspects (RQ10), the role service decomposition plays in distributing capabilities or computation (RQ11), and the role of AI/ML in crafting novel services or improving the service composition/decomposition process (RQ12). Other Technical sub-aspects not discussed in this SLR include service discovery and selection.
•**QoS Aspect**: The taxonomy adopted in this survey considers QoS as a full-fledged aspect of the IoT capabilities composition and decomposition topic. One reason is that, for example, as more capabilities are composed into value-added services and applications, concerns such as privacy (RQ15) are of great concern because composition formulas and preferences might give away stakeholders' personal preferences [218][219]. In addition to privacy, scalability (RQ13) and interoperability (RQ14) are two QoS sub-aspects that we will address as they relate to the identified research questions.

As indicated above, some Formal (composition algorithms, etc.), Technical (service selection, service discovery, etc.), and QoS (security, cost, energy efficiency, fault tolerance, response time, etc.) sub-aspects are not discussed in this SLR as the main topic or research question; they are represented in the taxonomy as "Other (Formal, Technical, QoS) Aspects." These other sub-aspects were either sufficiently discussed in the previous SLRs or were outside of the scope of this SLR. One goal of this SLR study is to encourage researchers to use and be inspired by the proposed taxonomy to build a comprehensive picture of IoT or CPS service composition and decomposition.

2) Manuscripts role and distribution

In the following subsections, tabulated results extracted from the primary studies -related to the research questions presented earlier- are exposed. This effort cites 257 items, including :

• 182 primary studies that met the inclusion/exclusion/filtering criteria as highlighted in Figure 9. These primary studies are referenced in column **Ref** from Table 4 to Table 18.
• 16 references to SLR and Literature reviews that represent the related work; however, these papers do not figure in this section's comparative analysis and discussion, nor were they used to provide answers to the SLR questions.
• 15 references were leveraged to explain certain aspects of the SLR methodology or to introduce or explain certain topics.
• 32 references to GitHub repositories were cited for enriching the discussion around certain composition platforms implementations and projects, formal verification tools, and source code. GitHub repositories also include [257], which contains extra details about the primary studies leveraged in this SLR.
• 12 online references that point to certain IoT composition tools or formal verification software.

The results of this SLR search are presented in the form of tables, which would be instrumental in answering the SLR questions in the discussion section. The next three subsections (**B**, **C**, **D**) contain data extracted for each aspect and sub-aspect mentioned in the taxonomy and relate to the SLR questions we seek to answer. Each sub-aspect of the taxonomy is introduced, and we identify which data -or Table- answers which RQ while explaining the columns in the tabulated data and specifying whether these tables provide answers to more than one RQ.

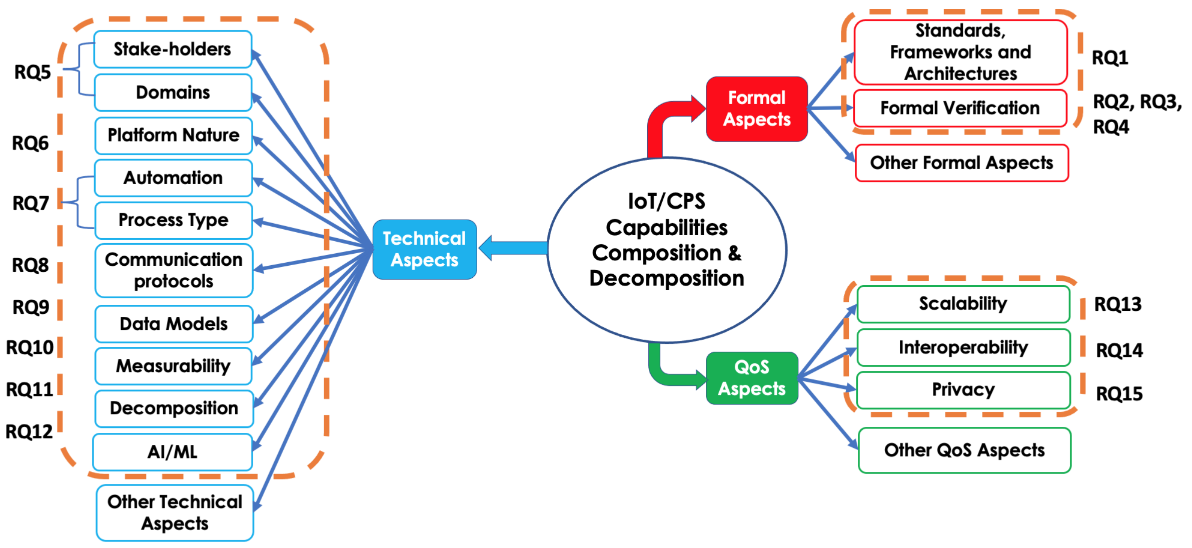

FIGURE 10. Proposed Taxonomy for the IoT/CPS capabilities composition and decomposition topic (root): Aspects (leaves), Sub-aspects (sub-leaves).

B. FORMAL ASPECTS

In this subsection, data related to the Formal sub-aspects of IoT/CPS capabilities composition and decomposition are extracted to answer RQ1, RQ2, RQ3, and RQ4.

First, the standards, frameworks, and reference architectures supporting the composition and decomposition concepts of IoT and CPS capabilities are addressed. Next, the key characteristics and implementations of the algebraic and graphical formal representations were analyzed. Formal verification techniques used to verify composite services, as well as tools and technologies that support such operations, are presented. Finally, the state-space explosion was given special consideration, with efforts and methods for solving this issue being discussed.

1) Standards, reference architectures, and frameworks

Standards, reference architectures, and frameworks provide foundations and guidance for building IoT or CPS platforms while respecting and guaranteeing certain aspects and constraints of interest to stakeholders. We list the different IoT and CPS capabilities frameworks that provide guidance on how to build composition environments with some criteria in mind [29]. Frameworks, standards, and reference architectures that propose composition/decomposition guidelines ensure that platforms built have certain beneficial properties (listed in column **Composition/Decomposition Enabled Properties** in Table 4), which would serve as a reason and motivation for the native support of composition/decomposition guidelines.

Table 4 shows primary studies that addressed this sub-aspect, and its data will be used to answer RQ1.

2) Composition algebras and formal representations

Algebra or formal representations can be used to shape algorithms for composition and can be leveraged as formal specifications in formal verification tools for assessing different properties of interest [178]. Formal descriptions of objects and their interactions, leading to composing and decomposing IoT capabilities, are performed using algorithms that rely on the algebraic representations of objects and services. Algebraic representations can also be derived from graphical representations using conversion [63].

In this paragraph, data that would help partially answer RQ2 is extracted, that is, recognizing the main properties of formal representations leveraged in service composition. Table 5 presents the efforts that have discussed formal representations and composition algebra. The data extracted from Table 5 provides an idea of how these formal representations are leveraged, whether graphical or algebraic in nature, and their important characteristics. For completion, a column for the source code related to composition algebra was provided to help the researcher use and explore implementations and use cases of these formal representations.

3) Service composition formal verification aspects

Formally verifying composed IoT services properties is a mechanism that aims to verify the properties of atomic or composed IoT capabilities using models in the format of formal specifications and running these models in tools to verify certain properties (deadlock freeness, correctness, fairness, etc.) [26]. The formal verification process is performed after the capabilities specifications are sketched to describe composed services using compatible composition algebra. The capabilities of IoT objects are typically described using a data model. This model is then converted into an algebraic language, which is translated into a formal specification language supported by a formal verification tool. The formal specification is later subject to a formal verification technique (model checking, equivalence checking, theorem proving) to verify -using a formal verification technique - that a certain property is met. The formal verification workflow for service

TABLE 4. Composition and decomposition aware standards, frameworks, and reference architectures

Standards, frameworks, architectures	Ref	Composition/Decomposition Enabled Properties
IoT Architecture (IoT-A)	[29]	• Native support for composition engines: FIWARE Functionality Groups (FG) & The Management FG.
Architecture Reference Model (ARM) and Web Ontology Language (OWL) based Frameworks	[35] [36] [220]	• Compatible with hierarchical and distributed systems. • Supports the conversion of OWL descriptors to Domain Specific Languages via different tools. • Assimilate composition and decomposition mechanisms to programming paradigms: classes and subclasses.
On-The-Fly (OTF) Computing Reference Architecture	[37] [221] [222] [223] [225]	• Brings on-the-fly automatic service composition to IoT platforms. • Simplifies composition rules regardless of platforms complexity.
NIST CPS Framework	[38] [39] [40]	• Defines service composition requirements in IoT and CPS environments. • Supported composition requirements: adaptability, complexity, constructivity, service discoverability, and selection.
Service-oriented service of cloud manufacturing (CMfg) CPS	[41]	• Inherits the NIST CPS Framework properties and integrates them into the OneM2M platform. • CMfg enables the composition of trustworthy and large-scale industrial cloud applications.
Internet of Smart City Objects (ISCO)	[42]	• Performs service composition with SCOs that satisfy functional and qualitative requirements at runtime. • Adaptable, flexible, and suits different composition contexts in a wide range of applications.
IoT and CPS Composition Framework (ICCF)	[193]	• Leverages the NIST CPS Framework composition guidelines. • Exploits the mPlane semantics to describe composition operations. • Relies on formal verification tools such as TLA+ to verify the composition models.
Web Service Decomposition Architecture	[252]	• Enables computation decomposition of complex and computation-intensive services from the cloud to edge nodes.

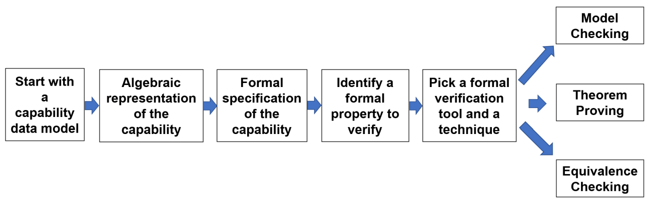
Start with a capability data model
Algebraic representation of the capability
Formal specification of the capability
Identify a formal property to verify
Pick a formal verification tool and a technique
Model Checking
Theorem Proving
Equivalence Checking

FIGURE 11. Formal verification of IoT/CPS capabilities workflow.

composition is illustrated in Figure 11.

In this paragraph, data that would help partially answer RQ3 is presented, that is, recognizing formal verification techniques, applications, and tools leveraged in service composition. This would also help in understanding the extent of the use of these techniques in research, service prototyping, or industrial applications. Table 6 aggregates primary studies that would contribute to answering this question; it provides data related to formal verification techniques, the properties they address, their domains of application, and the tool used to perform formal verification.

4) State Space Explosion

As the number of state variables in the composite system increases, the size of the system's state space increases exponentially, which makes it challenging to formally verify composed systems' properties. This is called the "state explosion problem." Much of the research on model checking over the past 30 years has involved developing techniques to address this problem [187].

Any composite system can have a large number of states. The size of the state space of a composed IoT system tends to grow exponentially as a function of the number of its capabilities, processes, and variables. The base of exponentiation depends on the number of local states of a capability or a variable and the number of values a capability or variable may store [28]. State-space methods have motivated researchers to efficiently reduce the number of states while remaining faithful to system design.

Previous surveys did not give this topic the attention it deserves and classified it as an open research problem [5].

Table 7 aggregates the efforts that tackled the problem of state space explosion in service composition and based on which an answer for RQ4 will be provided, i.e., the different methods for resolution and the extent of success of such methods in solving the state-space explosion problem in service composition.

C. TECHNICAL ASPECT

In this subsection, the Technical sub-aspects of the composition and decomposition of IoT capabilities are discussed. Domains of application, stakeholders' concerns, and real-world implementations (e.g., AWS GreenGrass + Lambda [126][110][28]) or efforts that provide substantial code-base or interesting prototypes (e.g., MCC Cloudlets [30]) were explored. IoT platforms for service composition and related communication protocols, data models, schemas, and engines will be discussed. The composition process types and automation, as well as the measurability of the novel

TABLE 5. Formal representations and composition algebras.

Algebraic/Graphical Representation	Ref	Nature	Description and relevant aspects	Src Code
Iterative Weighted Relaxation Service Composition (IWRSC)	[13]	Hybrid	● IWRSC algebra uses a directed graph for modeling services and operations ● IWRSC is used to model power consumption inefficiency in large-scale IoT environments.	N/A
Real-Maude	[21][59][74]	Algebraic	● Real-Maude is used to reason about real-time systems and interactions in terms of time. ● Real-Time Maude tool supports LTL model checking commands.	[75] [99]
LOTOS/ LOTOS New Technology (LNT)	[26][27][49] [54][55][65]	Algebraic	● Algebraic modeling style of IoT objects, interactions, states, and actions. ● This simple modeling enables the verification of nondeterministic and concurrent systems.	[66][67] [186]
Temporal Logic of Actions (TLA) / PLUSCAL	[28][193]	Algebraic	● Model checking of composite services using TLA after PLUSCAL conversion. ● The TLA+ tool converts the PLUSCAL model to a TLA specification.	[88] [240]
HTN-MLS (Hierarchical Task Planning for Machine Learning Services)	[32][33]	Algebraic	● HTN-MLS is an algorithm for automated service composition applied to the area of ML. ● HTN-MLS recursively decomposes complex tasks into subtasks until only atomic tasks remain.	[73]
Recursive Composition Algebra (RCA) and interaction graph (RCIG)	[43] [44]	Hybrid	● RCA yields a directed tree with a root service representation which allows traceability of services in distributed systems. ● RCA traces recursively services that compose a complex application in distributed systems built as a tree-leaf-root model.	N/A
DX-MAN (Distributed X-MAN)	[45][46] [47][237]	Hybrid	● DX-MAN is based on X-MAN, a component-based system modeling tool. ● DX-MAN is suited for specifying multi-workflow services during runtime.	[71]
Markov Decision Process(MDP)	[48][184]	Algebraic	● MDP is used on top of FSM and Probabilistic Computation Tree Logic(PCTL) to model formal properties such as reliability and cost. ● MDP uses states, actions, and rewards concepts to model discrete-time stochastic processes.	[243]
Finite State Machine (FSM)	[51][50]	Algebraic	● FSM enables the modeling of composite system states and the transitions between these states. ● FSM is especially suited for deterministic, interoperable, and complete systems.	[72]
Pi-Calculus	[52][185]	Algebraic	● Pi-Calculus is a refined classical logic that provides a method for tracking resources. ● Linear logic is leveraged to represent non-functional attributes, including cost and price.	N/A
Communicating Sequential Processes (CSP)	[56][57] [214]	Algebraic	● CSP is a mathematical theory for specifying complex patterns during concurrent interactions.	[76]
Extended Control Flow Graph (XCFG)	[58]	Graphical	● XCFG is an extension of CFG, which adds concurrency and synchronization dependency to model workflows of composite web services. ● XCFG models BPEL workflows and ensures synchronization among concurrent activities.	N/A
Business Process Model and Notation (BPMN)	[61][68]	Graphical	● An Energy Efficiency Algorithm (E2C2) based on the BPMN graphic formalism was used to model an event-based choreography of decoupled microservices compositions.	[69]
Directed Acyclic Graph (DAG)	[62][182]	Graphical	● Compositions are modeled using the DAG as a chain of services invoked successively. ● The result of the execution of one service invokes the next one.	[70]
Petri Nets/Colored Petri Nets (CPN) with a Kripke specification.	[63]	Hybrid	● CPN is a concurrent (as opposed to single-threaded FSMs) model. ● CPN is converted into an algebraic specification (Kripke) to describe and model-check a customer service system model. ● The Kripke specification is used to verify the reachability from and to other system states.	[77]
Vector Symbolic Architecture (VSA)	[64]	Algebraic	● VSA enables the compression of large volumes of data into a fixed-size vector. ● VSA hierarchically models composite features which can be decomposed into atomic vectors.	N/A
Calculus of Communicating Systems (CCS)	[78]	Algebraic	● CCS includes primitives for describing parallel compositions. ● CCS preserves synchronization and parallelism properties when converted to an LTS. ● CCS evaluates the qualitative correctness of properties such as a deadlock or livelock.	[79]
Process Meta Language (PROMELA)	[98][238]	Algebraic	● PROMELA was used in this example to model and check the properties of Advanced Electric Power Grids such as noninterference. ● RT-SPIN tool checks the correctness of the PROMELA model and determines whether it encounters the state space explosion problem.	[239] [97]
GALLINA	[244]	Algebraic	● Gallina, the specification language of Coq, was used to specify and prove distributed services mathematical theories based on building blocks including axioms, functions, etc.	[83][91] [246]
Intelligible semi-automated reasoning(Isar)	[245]	Algebraic	● Theorem proving of axioms using the proof language Isar within the Isabelle tool. ● Isar is known for its easy readability by humans and machines.	[85][90] [247]

TABLE 6. Formal verification techniques (Model Checking: MC | Equivalence Checking: EC | Theorem Proving: TP), verified properties, applications, and tools.

Ref	Techniques	Verified formal properties	Applications	Tool
[21]	MC	Deadlock Freeness + Unmatched Sent Messages	Mozilla Project Things Smart Home application	Maude
[26]	MC	Deadlocks+Livelocks+Safety+Liveness+Fairness	Majord'Home: SDN-based Smart Home platform	CADP
[27]	MC	Compatibility + Deadlock Freeness + Correctness	Majord'Home: SDN-based Smart Home platform	CADP
[28]	MC	Correctness	Fault-tolerance checking and bug detection in AWS systems	TLA+
[41]	TP	Trustworthiness	OneM2M NIST-CPS Framework requirements verification.	Mobius
[48][80]	MC	Reliability	Probabilistic model checking formal properties of IoT services	PRISM
[53]	MC	Deadlock Freeness + Correctness	Light control in Smart Homes	CADP
[60]	MC	Correctness	Service Composition in Multi-Cloud Environments	NuSMV
[81]	MC	Safety	Model checking safety and states of an air conditioner system.	CLEM
[82]	TP	Correctness + Reliability	Multi-stage composition formulas in smart health systems	Coq
[84][245]	TP	Correctness + Security	Privacy-Oriented Smart Health Application	Isabelle
[86]	MC + EC	Realizability	BPMN 2.0 Choreographies	CADP
[87]	EC	Bisimulation	An Intrusion detection system BPMN model specified via LTS	CADP
[94]	MC + EC	Deadlock Freeness + Liveness	Sequence Diagrams and Pi-Calculus Comparison	MWB
[92][244]	TP	Correctness	CPS Designs Formal Verification	Coq
[96]	MC	Compliance	NetBill Communication Protocol	CWB-NC
[98][238]	MC	Correctness+Deadlock Freeness+Liveness+Safety	Formal verification applied to MQTT-CV/CPS Applications	SPIN
[100][105]	MC	Correctness+Reliability+Consistency+Completeness	BPEL Processes and events in Web Services	Maude
[102]	MC	Correctness + Security	Verifying IEEE 802.11i correctness and security mechanisms	UPPAAL
[104]	MC	Correctness + Reliability	Online Ticket System represented as a Process Algebra	MWB
[106]	MC	Correctness	Online Book Purchase System	CWB-NC
[107]	MC	Correctness	E-Health	CADP
[108]	MC	Compatibility	Distributed Systems	TLA+
[109]	MC	Liveness + Safety	TLA+ Specification and Applications	TLA+
[110]	MC	Correctness	Hotel Room Reservation System	TLA+
[111]	MC + TP	Correctness	CADP Specification and Applications	CADP
[112]	MC	Safety	Cross Road Smart Transportation System	UPPAAL
[113]	MC	Correctness	Online Book Purchase System	MWB
[114]	MC + EC	Reliability + Compatibility	Addressing State Space Explosion in Petri-Nets services	Multiple
[115]	MC	Security + Privacy + Reliability	Formal verification of composed IoT Services properties	CompoSec
[116]	MC	Correctness + Reliability	Verifying a SysML Model after ACME/ARMANI conversion	AcmeStudio
[235]	MC + TP	Safety	Model checking non-functional properties of software systems.	TLA+
[236]	MC	Correctness + Safety + Security	Model checking functional/non-functional properties of IoT.	NuSMV
[241]	MC	Dependability	Model Checking industrial CPS properties	nuXmv

capabilities, are investigated. Novel Technical sub-aspects, including the decomposition of capabilities and the use of AI/ML in composing smart services or improving the composition process, are key contributions of this subsection. This subsection provides data that answer the technical questions RQ5 to RQ12.

1) Domains and Stakeholders

The applications of IoT composition cover multiple domains, including cities, buildings, transportation, health, farming, and manufacturing [121][23][41]. Different stakeholders have diverse expectations and requirements regarding building or leveraging composite capabilities. These stakeholders include end-users[117], developers[16], and city managers[120], to mention a few. Table 8 aggregates the efforts that addressed domains of applications of the ca-

TABLE 7. The state space explosion problem in service composition: description, techniques for resolution, and outcomes.

Ref	Description	Resolution Techniques and remarks	Outcome
[44]	• Modeling and verification of composed IoT services	• Trace Merging in Recursive composition algebra (RCIG) partially solved the state space explosion problem by reducing the order of the (RCIG)	Reduced
[49]	• Composite systems parallel model-checking and property verification.	• CADP Evaluator: proven to prevent a state-space explosion by enabling the detection of errors in systems with a large state space	Eliminated
[54]	• State-space explosion in IoT capabilities composition	• CADP toolbox: breaks the verification process into simpler verification problems	Eliminated
[56]	• FDR Models specification scalability	• FDR2 specification proposed: yielded fewer states, thus contributing to better scalability of the model	Reduced
[58]	• Shortcomings study (including state space explosion) of BPEL and Petri Nets specifications of concurrent systems.	• Limiting the size of the specification is a proposed solution to the state space explosion problem	Reduced
[96]	• GCTL introduced an improvement to the Computation Tree Logic (CTL) specification language.	• CWB-NC model checker: alleviates the state explosion problem of automata-based techniques. • The space requirements for Boolean functions used in the symbolic technique are exponentially smaller than those that use explicit representation. • The proposed technique cannot eliminate the state explosion problem because the state space still increases when the model becomes larger	Reduced
[110]	• Composing and verifying TLA specified composite services	• TLA's model checker TLC: equipped with a multithreaded concurrent verification mechanism that alleviates the state-space explosion problem • TLA+ tool allows offloading computation to AWS EC2 instances, which provides more resources to alleviate the state space explosion problem.	Reduced
[164] [189]	• State-space explosion in model checking for service composition models	• ML algorithms: applied to find the optimal service composition. • For future work, prediction methods such as deep learning will be leveraged to avoid the state-space explosion in model checking for service composition models.	Reduced
[190]	• Errors during microservice choreography composition in Cyber-Physical Social Systems (CPSS)	• The asynchronous compositions increase exponentially with the size of the simulator buffers. • PAT Simulator: stops when the number of states exceeds the simulator buffer.	Managed
[188]	• State-space explosion in cryptographic IoT protocols • Specifications of these protocols are not described in simple rules	• Proverif tool: its pi-calculus algorithms efficiently simplify the protocol's specifications (unification)	Eliminated
[238]	• The state space of the model of a CPS (Smart Grid) was large and could not be verified using the available computation resources.	• The model was decomposed into multiple sub-models, each with a smaller state space that can be checked individually.	Reduced

pabilities composition or decomposition and highlights the stakeholders' interests in each domain for each use case. Data in Table 8 are instrumental in answering RQ5, i.e., Understanding the major stakeholders' concerns regarding composing capabilities in different domains.

2) Composition platforms, engines, and implementations

Composition platforms addressed the composition and decomposition of IoT capabilities at different complexity layers, including edge[30], fog[122], and cloud[60] layers. The takeaway that can be concluded by studying composition and decomposition in these layers is the fact that the complexity of a service increases when its atomic capabilities are composed of edge devices, creating a fog service, or composed of fog services to create cloud services. However, this same complexity can overwhelm the upper layers, particularly the cloud layer [119]. Offloading computations through decomposition from the cloud nodes to the fog nodes or even to edge devices can prove necessary to perform capabilities in the most computationally-efficient manner. Table 9 classifies platforms based on the composition engine, the nature of the platform (simulation, centralized, decentralized), as well as the composition layers targeted (edge, fog, cloud), and provides the reader with implementation details and source code references. The data in Table 9 were used to answer RQ6, that is, understanding the technical differences in capabilities composition implementation in different platforms.

3) Composition Process Type and automation level

The process type refers to how the composition is triggered or processed and what its workflow looks like. Services can be composed in a serial [26], parallel [16], rule-based [147], or flow-based fashion [32], among other process types. Composition is a complex process that involves several steps, including discovery, selection, filtering, composing, verifying, and delivering. Some of these steps can be automated or semi-automated (requiring the input of a user or another entity). Similarly, each composition process has a set of tasks,

TABLE 8. Capabilities composition domains and corresponding stakeholders concerns.

Domain	Ref	Description/Application	Stakeholders' Concerns/Interests
Smart Buildings	[23] [27][26] [21] [117]	● Composed smart building services and scenarios. ● IoT Composer: smart building services composition. ● Automatic temperature/light management in a smart building. ● Person detection in a smart room	● Users expect a highly automated composition process. ● Users require ease of use of the platform. ● Ranking and filtering facilitate the selection of composition abstractions that are intelligible to users. ● User-friendliness of the platform's GUI.
Environment Monitoring	[51]	● FSM model-driven service composition architecture for the rapid prototyping of IoT services. Environment monitoring stations were implemented using three types of wireless sensor networks and deployed on a university campus.	● Developers and researchers can customize and quickly prototype composite services. These stakeholders also showed interest in cost reduction, reusability, and cross-domain interoperability.
Smart Transportation	[118] [119] [16] [64] [120] [197][199]	● UCEF, a co-simulation environment, allows the composition of features of complex systems such as autonomous vehicles. ● A platooning feature is composed: a technique that enables cooperation during adaptive cruise control for a series of vehicles. ● Dracena was used to compose vehicle sensor data for decision support and to predict insurance quotes based on thresholds. ● Traffic congestion at the city level uses Node-RED for a distributed implementation, with input from a traffic camera API. ● An estimator for parking, traffic, and noise was composed via FIWARE to provide an open trip planner. ● A travel booking/reservation composite service based on service-oriented computing architectures or web-based architectures.	● Assessing the trustworthiness of the functions of a CPS ● Road users expect improved safety and mobility. Developers benefit from the time efficiency gained by shifting from hardware and communication composition complexity to the ease of microservices composition. ● Insurance companies use Dracena to generate custom, fine-grained recommendations on insurance fees. ● Developers benefit from improved collaboration when Migrating the Node-RED implementation from a centralized to a decentralized paradigm. ● City planners seek cost optimization, thereby encouraging developers to build modular designs for their applications and identify reusable IoT services. ● Users benefit from the ease-of-use and plug-and-play automated features of the composition platform.
Smart Farming	[121]	● Gaiasense offers novel, inexpensive composite smart farming services by facilitating data interoperability for smart-farming systems using techniques such as "Data Interoperability Zone" and "Information Management Adapter."	● Farmers' interest lies in the zero cost of implementing Gaiasense. Producers of farming systems benefit from the reduced manufacturing cost. The environment also benefits from systems cooperation as it can reduce fossil fuel emissions.
Smart Cities	[122] [123] [20] [165][178] [179][180] [216]	● Application: FogFlow-based anomaly detection of energy consumption in a smart city. ● Adaptive service composition framework that supports dynamic reasoning. This allows mobile users to perform their daily tasks dynamically by integrating the services available in their vicinity. ● Smart city services include human-centric mobility, multimodal transport (parking, disabled people's navigation), and community policy applications (noise monitoring, agile governance). ● Energy-efficient IoT service composition algorithms are discussed with a focus on data sharing, fog-enabled, and mobile-based IoT applications. ● Smart-city composition platforms are discussed along with different impacted domains and design challenges.	● FogFlow's model allows IoT service developers to program elastic IoT services easily over the cloud or the edge. ● End-users discover and investigate more composition opportunities owing to the dynamic reasoning support of wEASEL-based composite heterogeneous systems. ● Developers are interested in publishing atomic services -leveraged by small companies to speed up composition- in a one-stop-shop repository. ● Energy-optimized composite services -achieved by adopting resource-efficient platforms (FSCA-EQ,..) and efficient composition algorithms (CRIO,...)- are the main concern for different stakeholders. ● Users are concerned about the composition platform performance, environment friendliness, security, low cost, and reliability.
Smart Health	[119]	● IoT capabilities were re-conceived as a "microservice" in contrast to the "thing" concept. This allows IoT to benefit from features such as the distribution of services and service discovery. The approach is illustrated in a personal health management service.	● Developers access ready-to-compose distributed, and secure microservices features using API Gateways. Users benefit from the enforcement of access control to composite microservices, which improves privacy protection for the data owner required for smart health applications.
Smart/Cloud Manufacturing	[41] [124] [125][181] [203]	● A large-scale composition platform for cloud manufacturing is introduced by connecting multiple remote factories and establishing a collaborative connection through the cloud. ● QoS stability algorithms were proposed to minimize supply chain manufacturing errors and, as a result, improve the competitiveness of the manufacturing facility. ● The difference between traditional manufacturing and service-oriented manufacturing was explained, and the benefits of cloud services in enabling collaboration and fault-tolerant composition of services between different providers were highlighted. ● SOCRADES, a smart manufacturing architecture, is presented. It relies on smart connected objects and tagged raw materials to build services that enable agile manufacturing of goods that accommodate customers' needs.	● This composition paradigm uses the NIST CPS framework to address the concerns of trustworthiness/stakeholders' (i.e., functional, human, timing, interoperability, and intelligence). ● Users and developers requirements: QoS stability, service collaboration, and service composition failures are discussed within a cloud manufacturing application. ● Recommendation algorithms are proposed to allow customers to compose products online based on their concerns (time, cost, reliability, availability, and throughput) in a fault-tolerant fashion. ● User-friendliness, ease of use, and customization are the major user concerns related to the proposed smart manufacturing composition platform.

TABLE 9. Composition platforms, engines, and implementations.

Platform	Ref	Platform nature	Composition Engine	Implementation Details	Src Code
mPlane	[12]	Microservices distributed as nodes (Consumer, Producer, Registry, Supervisor, Reasoner)	The **Supervisor** aggregates the lower-level capabilities into higher-level capabilities.	Implementation is available in Python3 [134], or NodeJS [135]. Requires a supervisor for composing capabilities, a capability source, and a requesting client. These client-server implementations can be run through an IP architecture or over a CDN.	[135] [134]
Fast IoT Composition	[13]	Distributed IoT services composition platform based on user-defined QoS requirements	The **Service Composition Process** composes atomic services based on various QoS needs.	Implemented using Java under a Windows 7 OS, and machine specifications were provided. No Git repository or reference to code is provided.	N/A
IoT Composer	[27]	Web application with a GUI for composing capabilities and verifying their correctness.	The **Majord'Home** platform and the **CADP** correctness verifier.	Web application hosted on Apache Tomcat. API uses jQuery and Semantic UI; calls are done using the REST API, serializing JSON objects	[143] [89]
FIWARE	[29]	Open-source decentralized/scalable IoT platform for data management and composition of smart applications.	The IoT Broker hosts the **Entity Composer Plugin** that composes services by aggregating attribute values.	Using the FIWARE composition plugin (iotbroker, entitycomposer) requires installing the IoT agent and enabling the entity to be composed.	[144] [145]
MCC Cloudlets	[30]	The platform provides composition at the edge via mobile devices, fog (virtual representations), and the cloud.	The **Central Cloud** composes Virtual Device Representations at the fog into applications	Networking between edge, fog, and cloud is ensured by OpenStack. Linux and android containers host virtual device representations residing in the fog. The Src Code column points to the platform's UPPAAL model and to the UPPAAL tool used for modeling.	[194] [101]
IFTTT	[37]	IFTTT is a cloud-based app available on iOS and Android. It provides access to public and private APIs.	**IFTTT** sets responses for events, connects to service providers, and executes commands.	Install the application on a supporting device and connect to APIs of interest. IFTTT applications include remotely controlling devices when a condition is met.	[139]
Home Assistant	[51]	Web-based home automation software for central control of smart home devices.	Home Assistant **Scripts**, **Scenes**, and **Automations** compose services similar to IFTTT's.	The home assistant software comes in different flavors. The most stable one uses an always-on VM that comes pre-installed and requires network configuration only.	[142]
Node-RED	[64]	Flow-based Web tool for composing services and connecting hardware and software services.	**JavaScript Functions** are inserted from a web interface to compose capabilities.	Install Node-RED using npm. On the web interface, drag and drop features from different APIs. Integration with IBM Watson's data analytics tool is available.	[138]
UCEF	[118] [136]	Simulation framework, which provides tools for creating simulations for CPS	Simulates a CPS via a federation: a composition of federates that interact via the **HLA Protocol**.	A template federation for a CPS is created using WebGME. The capabilities of each federate are populated using an IDE. ADS example in [118].	[137]
AWS Greengrass	[126]	A commercial cloud solution that provides various services, including storage and computing.	**Greengrass** is a solution for connecting and composing IoT services using **Lambda** scripts.	Install AWS Greengrass dependencies, certificates, and software for target devices. Leverage Lambda scripts to automatically compose services.	[127]
Microsoft Azure IoT	[128] [129]	Microsoft cloud service for building, testing, deploying, and managing services.	**Azure IoT Hub** enables bidirectional communications and composition between IoT devices.	Setting up an Azure IoT hub requires an Azure account, Azure portal, Azure IoT Tools, Azure PowerShell, Azure CLI, Azure REST API, and .NET templates.	[130] [131]
Cloud CAMP	[132]	An open-source cloud platform that automatically delivers composite applications in the cloud	**Abstraction of Business Model**, **Configurator**, **Enactor**, and the **Knowledge Base**	WebGME MDE is used to define the metaModel, MongoDB is used to store the model, and MySQL is used to store the knowledge base.	[133]
Vert.X	[140]	Message-driven toolkit for creating reactive, elastic, resilient, and responsive microservices	**Composition functions** implemented within Vert. X microservices.	Install Vert.X, and use its REST libraries (e.g., Axios) to connect to sources of data. The code references a well-being application that uses the Vert.X toolkit.	[141]
Thing Orchestration	[191]	A composition platform geared toward centralized computing architectures such as the cloud.	The **Orchestrator** is a central component coordinating a concurrent sequence of things to call during composition at runtime.	Python scripts were used to model a centralized architecture with a pool of resources with services to be composed. No Git repository or a reference to code is provided	N/A
MMESCN	[242]	A service composition **M**echanism based on collaborative **M**obile **E**dge **S**ervers and **C**ache **N**odes (MMESCN).	**Mobile Edge Servers (MES)** compose capabilities provided by IoT devices. **Cloud Center Nodes** take over when MES are not capable enough.	Network Simulator 3 (NS3) tool is used to simulate networks and edge or cloud composition nodes. One cloud center node is connected to 8 mobile edge servers. Each mobile edge server is connected to 20 IoT devices.	N/A

each with a certain level of automation. Different efforts have tackled the composition process type and automation level independently; in this effort, we try to find synergies and relationships between these two Technical sub-aspects. Table 10 highlights efforts that addressed composition process types and the automation level as well as manifestations for each topic. Data in Table 10 will be instrumental in answering RQ7, i.e., understanding how composition processes differ in terms of the automation level.

4) Communication protocols in service composition

From a communication perspective, capabilities composition and decomposition platforms adopt multiple communication paradigms and play different roles that vary depending on various contexts, and factors, including power consumption [24] or the environment type (production[27], simulation[136], ..), among other factors. Table 11 presents primary studies that tackled the communication protocols in service composition, their use cases, and their workflows. Table 11 data will be leveraged to answer RQ8, i.e., the role communication protocols play in the composition or decomposition of IoT/CPS capabilities.

5) Data Models or Schemas

The capability extracted from a device must be properly represented using a data model or schema to facilitate composition or decomposition. Efforts that tackle IoT capabilities composition typically tend to use data models based on known formats such as JSON, XML flavors, WSDL, HTML, and other options to represent IoT capabilities [150]. Device characteristics are described through ontologies that provide a shared vocabulary to model different objects and concepts, and their relationships [151]. Table 12 presents these efforts, and based on the results obtained from it, RQ9 is answered, i.e., understanding the roles of the different IoT data models and schemas leveraged in capabilities composition or decomposition.

6) Measurability

Atomic and composite capabilities differ in terms of measurability approaches, domains of interest, composed systems, and whether these capabilities are functional or QoS-related [15]. Table 13 contains primary studies that focused on measurability aspects of atomic and composite services; it also provides concrete composition examples and presents the capabilities measurability efforts based on the nature of the composite system, the measurable metric, the capabilities type, functionality state, and how capabilities are measured. Table 13 will also help address RQ10, that is, how can non-tangible composite capabilities be measured?

7) Decomposition

The decomposition of IoT services and capabilities is the process of deconstructing a complex service into small services or atomic capabilities [31][252]. This process contributes to cost-efficiency, scalability, and interoperability between IoT systems when complex services are built with decomposability in mind. Based on the research we performed, decomposition efforts are either i) capability oriented: which means a complex feature is decomposed into sub-features or atomic capabilities for reuse by other systems, or ii) computation oriented: a complex service might run in the cloud and consume more resources than allowed by a single service, decomposing computation allows its distribution on fog or edge devices in a way that renders resource consumption more efficient.

Decomposition is challenging because the building blocks of complex services might not be easily decomposable, especially when their APIs lack loose coupling support [119]. Decomposition also comes with nonfunctional challenges such as scalability, frequency of data generation, and security requirements. Table 14 aggregates service decomposition primary studies, and the data it contains will be leveraged to answer RQ11. i.e., exposing the benefits of building IoT platforms and complex services with decomposition in mind.

8) AI/ML

Researchers have addressed AI and ML use in service composition from two perspectives. The first perspective is capability-oriented [20], where the AI/ML capabilities are aggregated into value-added features with AI/ML capabilities. The second perspective is process-oriented [32]: AI and ML improve the composition process, especially service selection, where filtering criteria are considered for picking a particular capability over another. Table 15 illustrates the primary studies related to IoT capabilities composition that tapped into the AI/ML potential, and based on the data it presents, RQ12 is answered, i.e., understanding the role AI/ML techniques play in shaping novel capabilities or improving the composition process.

D. QOS ASPECT

This subsection investigates the important QoS sub-aspects of scalability, interoperability, and privacy in service composition or decomposition. A unified approach is followed for studying these QoS properties via the SLR questions RQ13, RQ14, and RQ15, respectively.
For each concern, the context of the study is presented, challenges encountered against its realization, and how researchers tackled this concern in their service composition efforts.
This subsection will provide input for answering QoS questions RQ13, RQ14, and RQ15 based on the tabulated results in Tables 16, 17, and 18.

1) Scalability

Scalability in the context of capabilities composition is the ability of a particular IoT composition platform, composition algebra, or a composition technical implementation, to function correctly regardless of the number of composite capabilities [45]. Table 16 shows the capabilities composi-

TABLE 10. Service composition process types and automation levels.

Composition Process Type	Ref	Process Manifestation	Automation Level	Automation Manifestation
Sync/Async Support	[12]	● The data and control flow within mPlane supports both cyclic workflows in the "foreground" and continuous/periodic measurements in the "background". ● mPlane interfaces facilitate the flow of control messages that trigger new measurements and data reception, supporting both synchronous and asynchronous modes.	semi-automatic	● mPlane is highly programmable and supports component-initiated workflows (no client interference required) and client-initiated workflows.
Flow-based and Programming-based	[13]	● The composition flow requires rendering constrained devices' capabilities into virtual objects before the composition and defining their QoS in a way that contributes to either their selection or exclusion. ● An Iterative Weighted Relaxation Service Composition (IWRSC) algorithm is proposed that allows users to customize QoS parameters that contribute to the selection of atomic capabilities participating in a composition.	semi-automatic	● Composite services are realized as a succession of tasks. ● Tasks represent abstract services that can be realized by the best-fitting concrete service. ● The user decides on the fitness of a service based on its QoS properties. ● QoS properties are either device-specific or user-defined; hence, they are semi-automatic.
Asynchronous or Parallel	[16]	● Dracena compositions enabled the asynchronous and cooperative use of large amounts of IoT data among disparate services in real-time.	semi-automatic	● The composition development environment provides Kafka plugins that perform compositions on basic and complex capabilities. ● It also provides service designers APIs to inject hypotheses and verify output correctness.
Rule-based	[21]	● Automated techniques for building compositions of devices represented as abstract objects are proposed. ● Filtering and Ranking rule out non-desirable compositions as the number of possible compositions is high.	automatic	● For the Node-RED deployment, human intervention is only required to define the goal of the composition. The other composition steps were then automated.
Flow-Based	[17] [64] [146]	● Node-RED, WoTKit, and VITAL-OS are flow-based IoT programming tools. In particular, Node-RED models composition task logic as a directed flow chart that consists of sequences of interconnected service nodes. ● Flow-based composition can be easily represented in GUIs, which increases usability and provides more intuitive user interaction. ● When it comes to complicated composition task logic, flow-based composition tools cannot avoid introducing additional programming.	automatic	● The WotKit Processing Engine automates data collection and processing using scripts. ● The VSA approach applied in Node-RED enables automated service composition by combining and self-describing services. ● VITAL-OS, an open-source operating system for smart cities, aims to automate smart city services in a three-phase process: infrastructure deployment, deployment of vertical applications, and deployment of city-wide integrated applications.
Synchronous or Serial or Sequential	[26] [27]	● Composition aims to connect various objects through their APIs. ● The objects involved in a composition interact through bindings in a synchronous manner.	automatic	● IoT Composer graphically exposes the interfaces of the selected objects and allows the user to bind interfaces concerning the IoT service expected from the composition. ● It is worth noting that only this first step of the approach requires human intervention; all the subsequent steps are fully automated.
Flow-based	[32] [33]	● MLS-PLAN: a Machine learning services planner that leverages an algorithm for automating machine learning data classification.	automatic	● The automated task composes an ML pipeline (consisting of ML services) that maximizes classification accuracy over new data from a data source
Process or Programming Based	[51]	● Programming or process-based composition is the most widely used method in which the business logic of the service is composed mainly by manual programming or using scripts. ● This method adopts either traditional programming languages or domain-specific languages.	semi-automatic	● This method allows the encapsulation of heterogeneous sensors, actuators, and IoT devices into composable Web services with uniform model-driven,user-customizable, semi-automatic compositions
Rule-Based	[147] [148]	● In openHAB 2 and IFTTT, rules are usually predefined and represented as events, conditions, formulas, or symbolic logic. ● Whenever a rule is met during runtime, the corresponding operation is automatically triggered.	automatic	● IFTTT and openHAB only require setting up simple rules via a GUI that trigger events in the context of home automation. This approach is deemed automatic owing to its GUI accessibility and low complexity level.

tions scalability challenges and the different ways researchers addressed those challenges. Data in Table 16 is extracted from primary studies that discussed scalability challenges in the context of service composition and provided solutions to those challenges. Table 16 will also help address RQ13, i.e., recognizing the main scalability challenges and adopting solutions to improve scalability in composition platforms.

2) Interoperability

Enabling the full compositionality potential of the future CPS and IoT platforms requires enhanced interoperability between software and hardware elements, supported by new reference architectures, standard definitions, and lexicons [173][121]. Addressing this challenge requires broad collaboration to develop a consensus around key concepts and

TABLE 11. Communication protocols roles in IoT/CPS service composition/decomposition.

Communication Protocol	**Protocol description**	**Ref**	**Protocol Use case / Workflow**
mPlane Protocol	• mPlane refers to a protocol in which entities request, receive, compose, and store data by leveraging multiple components.	[12]	• This example leverages mPlane as a measurement platform. Specifications are sent by a consumer to request measurements from probes capabilities.
Representational state transfer (REST/HTTP)	• A software architectural style that guides the design and development of web services	[14] [22] [160]	• RESTful communication technologies are used to connect the IoT server proxy to both the smart city controller (using HTTP) and the Smart Service Provider (using CoAP). • A solution for composing asynchronous RESTful web services, making use of various IoT services, invoking the RESTful web services from the IoT, and publishing a BPEL process as a RESTful web service by extending BPEL. • The proposed composition platform enables transparent access to heterogeneous IoT networks -including CoAP- using "interworking" proxies to the HTTP protocol.
multicast DNS protocol (mDNS)	• mDNS resolves hostnames to IP addresses within small networks that do not include a local name server.	[21]	• mDNS is used to enable the discoverability of IoT devices.
Constrained Application Protocol (CoAP)	• An internet application protocol (RFC 7252) that enables communication between constrained devices.	[24]	• SVOM (Service and Virtual Objects Management system) is introduced with a CoAP-based fault management component that captures the status of the registered devices and services.
Software-Defined Networks (SDN)	• A network management technology that enables dynamic, programmable, and efficient network configuration.	[27] [41]	• Majord'Home leverages a software-defined LAN (SD-LAN) for its simplified interfaces required for network configuration of IoT services involving multiple smart environments. • A testbed based on SDN is constructed for experimental verification in a flexible network configuration with Mininet as an emulator for configuring and deploying virtual devices
Simple Object Access Protocol (SOAP)	• A messaging protocol specification that enables the exchange of structured data in web services	[56] [159]	• The service requester and service provider exchange data using SOAP service request and response messages. • Genetic algorithms and case-based reasoning are leveraged to define automatic composition workflows in SOA and select suitable atomic services based on select QoS requirements.
Message Queuing Telemetry Transport (MQTT)	• A lightweight, publish-subscribe network protocol that transports messages between devices and usually runs over TCP/IP.	[122] [147]	• Fogflow's NGSI-based context management system provides a global view for all system components, enables querying, subscribing, and updating context entities, and supports MQTT pub-sub message brokers such as Mosquitto and Apache Kafka. • MQTT is used as middleware for its pub/sub-broker, which runs on top of a TCP/IP network for the openHAB automated home system.
High-Level Architecture (HLA)	• A standard for the distributed simulation used when building a simulation for a larger purpose by combining several simulations	[136] [118]	• HLA is used as a medium for exchanging data between federates composing CPS in UCEF simulations.
WebSocket	• A communications protocol capable of bidirectional communications over a single TCP connection (RFC 6455).	[140]	• Netty is compared with other WebSocket frameworks, including Undertow, Vert.X, Grizzly, and Jetty, to decide which situations each framework ought to be used for.
Named Data Networks (NDN)	• Named Data Networks (NDN) is an Information-Centric Networking (ICN) protocol that is seen as an alternative to TCP/IP.	[161] [162]	• NDN provides the benefit of node caching and service decomposability by tracing back its interest requests. • Improving NDN's high-speed forwarding capabilities using NDN-DPDK techniques offers faster forwarding speeds than known IP protocols. These benefits can be extended to other applications, such as IoT communications capabilities.
Wireless Sensor Networks (WSNs)	• Wireless sensor networks are dispersed sensors that track physical environment data and forward it to a central location.	[210] [211]	• Principles of the WSNs and SOA are exploited to enable energy-efficient composition features in the Networks On a Chip (NOC) paradigm. • WSNs and smart connected devices (IoT devices) are leveraged to develop new applications in a specific domain using the concept of "mashup" architectures or user-generated composite applications.
GS1 EPCglobal	• A set of protocols (RFID, GDSN) and standards (EPCglobal) that facilitate data exchange between different stakeholders in the supply chain domain.	[169]	• An EPCglobal-compliant IoT Middleware (Fosstrac) was used to validate a proposed IoT Infrastructure model which uses the Application Level Events (ALE) and the Electronic Product Code Information Services (EPCIS) components of the EPCglobal standard to support service decomposition, multi-threading, elasticity, and cloud virtualization capabilities.

build a shared understanding of the underlying composition technologies[256]. Interoperability in the context of IoT composition refers to the ability of a particular platform to compose capabilities from devices with different data models or that, in general, are not compatible or not built for straightforward composition.

TABLE 12. Capabilities data models/schemas properties, roles, and examples of attributes.

Data model/schema	Ref	Data model/schema role	Examples of data models attributes

XML	[25]
	[152]
	[206]
		• XML was used in mPlane for integration with XML-based systems.
		• RDF/XML provides APIs for backward compatibility with other platforms.
		• Describing and discovering services from different Wireless Sensor Networks (WSN) nodes and networks using XML and SOAP standards.
			RDF/XML schema attributes:
			rdf:statement;
			rdf:subject;
			rdf:predicate; rdf:object

WSDL	[44][201][209]	• WSDL is used with SOAP and XML schemas to describe web services through a document that lacks implementation details and logic, which is why WSDL is limited to modeling and not formally verifying operations.	WSDL schema parameters: Port Type; Operation; input; output
JSON-based schemas: JSON JSON-WoT mPlane-JSON JSON for linking data (JSON-LD)	[26] [27] [192] [21] [25] [152] [64]	• The JSON schema represents the objects, the bindings, steps, and strength of the bindings and the composition plan. • The JSON-Web of Things schema is used to describe IoT objects' capabilities and also the IFTTT rules that describe the synchronous compositions. • The mPlane reference architecture leveraged JSON as the default schema for its simplicity, parseability, and efficiency. • For the evolution of context information representation, JSON-LD provides more powerful options and is designed to be easily integrated into the existing Context Information Management (CIM) platforms. • JSON is used for a service description of the Node-RED object detectors. JSON service descriptions were later converted into semantically comparable service vector descriptions.	mPlane capability Attributes in the JSON schema: parameters start: now...+inf end: now...+inf source.ip4: 192.0.2.3 destination.ip4: octets.count: 28...65535 period.s:

YAML	[25]	• YAML's main use cases for the mPlane platform include documenting and debugging for its improved human readability and writability.	YAML's schema attributes :
			template; interfaceMap;
			activeTimeout;
			maxFlows;
			silkCompatible; ...

HSML/HTML	[51]	• HSML is a domain-specific language with HTML-like syntax for describing the state of an IoT service and the state transfer chain between services.	HSML Schema Attributes : loc (x,y), id, src (ip,protocol) type , filter
OWL-based schemas: OWL-S OWLS-TC4	[110] [183] [157] [123]	• OWL-S documents describe services in terms of predefined upper ontology, such as service profiles, grounding, and processes. A formal TLA specification is defined based on the OWL-S description. • IoT services are defined as a subclass of the service class defined in OWL-S. An IoT Service has a Service Profile and a Process that describes its functional and nonfunctional properties (inherited from the OWL-S Service class). • OWLS-TC4 was used as a data model to create composite services in the context of smart cities. Authors argued that data models such as OWL-S provide simple capability aggregation mechanisms for non-complex compositions.	OWL-S Service schema attributes: service name, port types, service operations, input message, output message, implementation descriptions

WoTDL	[149]	• The Web of Things Description Language (WoTDL) ontology is an alternative to existing WoT models such as OWL-S and WSDL, which are not suitable for describing AI planning concepts for automatic WoT composition.	WoTDL schema attributes:
			hasActuator; hasSensor;
			name; hasMeasurement;
			hasPreCondition;
			hasActuation; hasEffect

BPEL-based schemas:
BPEL
WS-BPEL
BPEL-TC
	[153]
	[154][156]
	[155]
	[205]
		• WS-BPEL is assimilated to UML as it allows the description of services and enables the description and execution of composite services.
		• Composite services based on WS-BPEL were tested under load in a travel-agency composition prototype.
		• BPEL-TC is an adaptation of WS-BPEL that accommodates composition and decomposition requirements for temporally customized web services.
		• BPEL process models are transformed into composite executable service templates after three steps (template creation, composition, and installation.)
			BPEL schema attributes:
			Travel Request; Invoke FS;
			resp FS; type input;
			Travel Response;
			Type output

Semantic Sensor Network (SSN)	[158]	• The SSN ontology (part of the W3C Semantic Sensor Networks Incubator Group) represents context information for sensor devices, including deployment attributes.	SSN schema attributes : available battery, deployment attributes, location, time

Table 17 aggregates efforts that addressed interoperability aspects, challenges, and potential solutions; as a result, it will help answer RQ14, i.e., understanding interoperability challenges and adopting solutions to improve interoperability aspects in composition platforms.

3) Privacy

For IoT/CPS to be trusted by different stakeholders, privacy must be preserved when sensitive and Personally Identifiable Information (PII) is exchanged [84]. IoT and CPS capabilities are typically associated with privacy requirements, especially

TABLE 13. Atomic and composite capabilities measurability.

Ref	Composed System	Measurable Metric	Capability Type	Functional ?	How Measured
[12]	ISP/CDN Collaboration platform	• Traffic Volume between end users and the CDN	Composite	YES	• Among mPlane's main roles is communication performance measurement. Hence the functional classification of the traffic volume metric. • Traffic Volume is provided by mPlane-wrapped SNMP interface counters and is based on Bandwidth and Latency.
[15]	Thermal Comfort System	• Thermal Comfort • Temperature • Humidity • Carbon Dioxide • Wind • RTT • Latency • Drop Rate	Composite Atomic Atomic Atomic Atomic Atomic Atomic Atomic	YES YES YES YES YES NO NO NO	• Thermal comfort is not measured by a single metric but rather assessed by end-users based on the atomic functional capabilities data (temperature, humidity, CO2, wind). • Atomic functional capabilities are measured using data coming from physical sensors (BME 280, MQ-2) • Latency and Drop Rate are measured using the communication probes within the REST communication model. RTT is the time between task generation and application response.
[16]	DRACENA's Composite Traffic Capabilities System	• Traffic Jam Trend Prediction (TJTP) • Driving Danger Map Generation (DDMG)	Composite Composite	YES YES	• The Traffic Jam Trend Prediction capability is composed based on speed and position atomic features provided by the SUMO simulator. • The Driving Danger Map Generation capability is generated using atomic SUMO features, including Wiper Data (rainfall level), braking data, and position.
[26][27]	Smart Home Composition System: multimedia, smart Control, ..	• Tool Usability • Performance	Composite Composite	NO NO	• End users input on clicking, binding, composing, and interacting with UI. The time spent on tasks is also considered. • Researchers assessed the composition performance based on the number of bindings, states of the system, time to generate a specification, and time for deployment.
[46]	Smart Cities Parking (SPark)	• Scalability of the compositions	Composite	NO	• Scalability is defined as the ability to handle increasing workloads in an IoT system over time. • Workloads are typically measured in terms of either the number of requests dispatched or the data streams generated.
[166]	Smart Buildings Echosystem	• Thermal Comfort • Energy Efficiency • Security • Tele-management • Visual Comfort	Composite Composite Composite Composite Composite	YES YES YES YES YES	• Sensing, interaction, and events are collected at the Data Layer. The data are processed through the Information Layer and the Knowledge Layer for data composition. • Composite services are served at the Services Layer with a metric for stakeholders to assess.
[167]	NFP-Aware Service Composition Platforms	• Response Time • Price • Reliability • Availability • Platform • Location • Provider	Composite Composite Composite Composite Composite Composite Composite	NO NO NO NO NO NO NO	• RTT or the Price (Quantitative) calculated as the aggregation of the delays (or prices) of the composite system's atomic services. • Reliability and Availability (Quantitative) are calculated as the product of the composite system's atomic services reliability and availability scores. • Platform, Location, and Provider (Qualitative) are composed by picking a user's favorite parameter.
[198][200]	Adaptive/dynamic Service Framework	• Service Quality • Network Capacity	Composite Composite	NO NO	• The service quality or network capacity is assessed using formulas that consider many atomic factors of a service, such as its resource utilization.
[212]	Web-services composition platform with a QoS probabilistic estimator.	• Performance • Resources • Dependability • Fidelity	Atomic Atomic Composite Composite	NO NO NO NO	• Performance assessed through the response time: time between submission of a request and reception of a response. • Resources assessed via the cost: the amount of money to execute an instance. • Dependability is assessed based on a composition of probabilities involving Reliability and Availability. • Fidelity is assessed through a reputation rating in the form of a scalar value.
[215]	Web-service composition in distributed cloud environments.	• Response Time • Availability • Price	Atomic Atomic Atomic	NO NO NO	• Time between sending a request and receiving a result in [ms]. • Probability of the availability of a service [%]. • Money, the applicant pays the service provider to use the service [$].

in applications that involve personal data collection; therefore, there is a need for anonymization techniques as well as encryption and secure data storage. In [174], privacy is also considered the key to maintaining the success of capabilities composition in the cloud and its impact on sharing information for social networking and teamwork on a specific

TABLE 14. Service decomposition efforts description and benefits

Ref	Platform	Decomposition Description	Decomposition Topic	Decomposition Benefit
[13]	Energy-efficient and fast IoT service composition	• Relaxation is a technique used to improve the flexibility and scalability of service composition. • Relaxation decomposes users' requirements and QoS constraints into local constraints of services.	Capability	• Decomposition points to energy-efficient atomic capabilities that compose energy-efficient services.
[31]	FOG-IoT and Linked-Microservices	• Linked microservices are built with decomposition in mind, contributing to the computation across different computing nodes in the IoT architecture and benefiting from fog and SOA strengths.	Computation	• Congestion reduced by (70%) at the cloud.
[51]	FSM model-driven service composition and decomposition architecture	• From a machine state perspective, composing or decomposing IoT devices can be described as building or breaking the linkage of states between FSMs.	Capability	• Reusability of FSM model-driven services.
[64]	Node-RED/VSA	• The proposed scheme converts existing Node-RED microservices into a cooperating set of decomposed and decentralized proxies. • These proxies are instantiated into the CORE environment by adding a cognitively aware wrapper around each service to facilitate decentralized discovery and execution.	Capability	• Decomposition decentralizes services and improves collaboration between capabilities.
[119]	Microservices-based IoT platform	• Containers leveraged in microservices to enhance safe service decomposition and enhance the security of IoT infrastructure. • Desirable characteristics of microservice-based systems include the decomposition of larger services into small, focused, self-contained services with loose coupling.	Capability	• The loose coupling allows fast decomposition and, as a result, faster reuse of existing capabilities for further compositions
[161]	Named Functions Networks Computation Service Management (NFN CS-Man)	• Decomposition is built into the information request of the NDN IoT network model. • Sending a request (interest) is later subject to decomposition to sub-interests, each requesting an aspect of the composite service.	Capability	• Interests retrieve cached capabilities from a nearby cache. • Traceability of atomic capabilities by checking the interests in the Pending Interest Table (PIT).
[166]	iKaaS functional decomposition	• Each simple service runs in its process and communicates using lightweight mechanisms. • The overall high-level service logic is decomposed into multiple software modules, delivered as independent runtime services.	Capability	• Service functional decomposition ensures agile, autonomous, flexible, and scalable services.
[168]	Fog-IoT ORchestrator (FITOR)	• Decomposition of application computation (CPU, memory y) on different nodes is performed e using optimized fog service provisioning g (O-FSP)	Computation	• Resources/cost optimization. • Stakeholder acceptance
[169]	EPCES-ALE	• Service decomposition is considered one of the pillars for a "Future IoT Infrastructure" along with virtualization and multi-threading in order to distribute computation optimally.	Computation	• Computation optimization
[170]	Hierarchical IoT	• A middle-layer service can be decomposed into sub-services which can be integrated to complete the pieces of another service with higher complexity.	Capability	• Reusing atomic capabilities from a complex service into another service
[171]	Decomposing monoliths into smaller microservices using domain-driven design (DDD)	• For monolithic software systems with a complex domain, decomposition faces many challenges, including the low comprehensibility of its source code. • Add to these challenges that every change in the system needs a whole redeployment, and all parts of the system are not equal in terms of change frequency. • DDD is a good option for the initial decomposition of a system as it can be applied to a subdomain to decompose it into smaller chunks.	Capability	• DDD ensures reusability through the decomposition of small chunks of a system.

project. One way to keep this success is to allow users to choose when and what they wish to share and to allow encryption and decryption facilities to protect specific data.

Table 18 is an aggregation of primary studies that discussed the capabilities composition privacy aspects, challenges, and solutions adopted to address privacy. Data in Table 18 will constitute a basis for addressing RQ15, i.e., recognizing privacy challenges that arise while composing capabilities in IoT platforms and understanding the different solutions the researchers adopted to improve privacy and tackle its challenges.

V. DISCUSSION

Section III describes the adopted SLR methodology, and Section IV presents the results with tabulated data that would

TABLE 15. AI/ML use in service composition/decomposition: description of efforts, orientation, applications, and mechanisms.

Ref	Description	Orientation	AI/ML applications and mechanisms
[20]	Using AI/ML atomic service to compose intelligent and complex capabilities in the smart-city domain (EU's SynchronyCity project).	Capability	• Composing ML complex capabilities, including parking area availability, traffic flow, and noise level predictions based on time-series data. • These complex capabilities will be used as atomic capabilities to build smarter and cross-city applications
[31]	Facial recognition service that is composed of ML tasks, including facial feature extraction, data fusion, data filtering, and face detection algorithms	Capability	• Services decomposition via fog computing was introduced to shift computation latency and burden from the centralized cloud to decentralized fog nodes regardless of data types (numerical, text, image) and the different ML algorithms. • Decomposition of facial recognition capabilities works with different ML algorithms and data types.
[32] [33]	Composing ML pipelines to improve classification outcomes	Process	• Given sample data, the task is to compose an ML pipeline (consisting of ML services) that maximizes classification accuracy over new data from the same source. • AI-based Classification can be leveraged in service selection, thereby improving the composition process
[115]	AI is used to verify nonfunctional properties of a composed system, including security, privacy, and dependability, and track its changes over time	Process	• CompoSecReasoner framework uses AI algorithms to monitor the composite system and its changes and evaluates its security, privacy, and dependability status. • CompoSecReasoner utilizes an event and model-based approach and implements dynamic system composition verification, properties validation, and automated administration based on metrics.
[120]	Reusable AI atomic smart-city services.	Capability	• AI-enabled smart city services were analyzed and compared collaboratively. This resulted in atomic services such as parking and traffic estimators that use AI. • These atomic AI capabilities can be used to enable more complex capabilities in a collaborative platform.
[159]	Genetic algorithms are used for service selection based on QoS defined by the user in SOA-based environments.	Process	• An approach to deal with the dynamic service composition problem is presented based on a genetic algorithm and case-based reasoning, which supports the flexible service workflow according to the user's requirements.
[163]	AI algorithms are leveraged for geospatial web service selection and automatic composition	Process	• Integration of well-described web services into feasible workflows can be achieved using AI planning, a computational deliberation process that chooses and organizes actions to achieve predefined objectives.
[172]	Service selection using an agent that uses reinforcement learning to select devices for composition based on specific user-defined criteria	Process	• A reinforcement learning technique was adopted to effectively deal with highly dynamic situations in mobile IoT environments. • A reinforcement learning agent was proposed that selects services in terms of spatial cohesiveness and number of handovers while providing the services.
[248]	Predictive maintenance in Industrial IoT platforms enabled by A Federated Learning Framework.	Process	• IoT platforms maintenance impacts not only nonfunctional properties such as the QoS but also process-related operations such as service discovery/availability. •Maintenance and fault data are collected over time and based on which a prediction is made to infer when the next fault will occur
[249]	Integrating and adapting Machine Learning for security monitoring in Big Data IoT data aggregation platforms.	Process	• Ensuring the security of IoT platforms through AI can be instrumental in guaranteeing certain composition processes are not compromised (service selection: a compromised atomic service might lead to compromised composite service.). • Phases for implementing security-aware machine learning processes within IoT Data aggregation platforms are explained.
[251]	Composing real-time AI capabilities within the edge layer microservices.	Capability	• The platform implements AI capabilities at the edge layer (called ROOF) and only calls upper layers (Fog, Cloud) for composition or computation-intensive tasks. • This platform was used to analyze incoming AC, Speed, and Fuel Consumption data from simulated vehicles with the goal of leveraging these time-critical atomic capabilities to build composite features in an automated driving scenario.

help provide answers (AQs) to the pointed-out SLR questions (RQs). In this section, **A)** the data from Section IV is analyzed and consolidated to provide answers to SLR questions RQ1-R15, and **B)** trends, gaps, and threats to the validity of this study are indicated.

A. ANSWERING SLR QUESTIONS

In this subsection, we provide answers to the SLR questions by referencing relevant primary studies while highlighting in figures trends for topics of interest.

1) AQ1: What is the motivation for native support for IoT capabilities composition/decomposition mechanisms by standards, reference models/architectures (RMAs), and frameworks?

IoT/CPS standards, frameworks, and reference architectures not only define data sharing, interoperability, and security specifications of exchanged messages over a composition platform but also incorporate mechanisms for composing or decomposing novel capabilities in different platforms.

The different standards, frameworks, and reference architectures mentioned in Table 4 provide motivations for native

TABLE 16. Scalability challenges and solutions in IoT/CPS service composition.

Ref	Context	Scalability Challenges	How Addressed
[26] [27]	IoT Composer scalability in smart homes	Formal verification scalability challenge: number of states.	• Formal verification: CADP scales well as it can handle hundreds of parallel compositions.
[38] [39] [40]	NIST CPS Framework Scalability Constraints	Billions of connected IoT devices present scalability challenges: (i) Network performance (ii) Authenticating large numbers of users	• Building IoT/CPS capabilities composition platforms with the NIST CPS Framework guidelines in mind, especially those related to trustworthiness, would make these platforms more scalable.
[45] [46] [47]	DX-MAN scalability compared to existing paradigms	Composing IoT systems from an ultra-large number of services	• Scalability requirements: (i) explicit control flow (ii) distributed workflows (iii) location transparency (iv) decentralized data flows (v) separation of control, data, and computation; (vi) workflow variability
[154]	Testing WS-BPEL compositions under load (Number of Requests).	An online travel recommender tool was used to test the WS-BPEL compositions triggered by the user requests; each represents a composition of user travel criteria.	• The system scalability and performance were satisfactory for small-scale compositions • For higher load compositions, load balancing was proposed as a solution.
[166]	IoT Big Data Analytics Platform	Variable number of devices, services, and users	• Data management, storage, and processing services need to be dimensioned dynamically
[175]	secureSVM: a privacy preserving Data Distribution Platform for ML Data Training and Composition	Training data originates from multiple providers. This can overwhelm training/composition algorithms.	• A Gradient Descent Optimization algorithm was adopted for training and composing data, making the platform scale with many data providers.
[207]	Composition algorithms that scale for services that capture qualitative user preferences.	Proving that using qualitative preferences can help improve the quality of the generated composition algorithms, including their ability to scale.	• Integrating CP-net-based reasoning with traditional approaches to composition would improve the scalability of the composition processes.
[250]	Wearable health devices' scalability constraints	(i) High cost of wearable devices (ii) When widely adopted, computing environments might not handle the large volume of generated data, especially when aggregation is required.	• Picking cost-effective devices. • The computing architecture consists of a web service that leverages Edge, Fog, and Cloud layers to achieve scalability with real-time and computation-intensive constraints.

support for IoT/CPS capabilities composition/decomposition through their enabled properties. The motivations include: a) the ability to address the composition of functional, business, human, trustworthiness, timing, data, boundaries, composition, and lifecycle concerns of an IoT or a CPS, and b) taking into consideration the complexity, discoverability, adaptability, and constructivity of the composite capabilities [38][39][40]; c) enabling computation distribution of computation-intensive services running in cloud nodes to low computation nodes at the Fog/Edge level [252]; d) addressing composite services functional and qualitative properties during runtime to enable flexible and adaptable compositions [42] and e) incorporating composition-friendly ontologies and composition mechanisms in distributed environments through hierarchical structures such as classes and subclasses [35][36][220]; f) providing automatic composition mechanisms to build modular software capabilities and from heterogeneous service marketplaces and locations [37] [221] [222] [223][225]; g) enabling reusability of small, atomic, reusable components through decomposition[252] [252][29]; and h) guiding atomic service discovery, selection, and complex services prototyping and composition in the cloud environments [252] [41] [40]. Figure 12 summarizes the motivations above.

2) AQ2: What are the main properties of formal representations leveraged in service composition ?

Based on data from Tables 5 and 6, formal representations have two main properties : the **a) Modeling** property, and the **b) Formal verification** property, given that the formal model can be used in formal verification tools, and as a result, the properties of interest can be formally assessed.

a) Modeling : can be performed in three ways: algebraic [26][57][59][65][74][184][185][214], graphical [61][62][182], or hybrid [63][47][45][46][237] . Modeling targets two elements:

$\rightarrow$ **Functional components**: include modeling system's components (e.g., modeling system buffers [86]), service types (e.g., describing and tracing atomic or composite services[189][13][43]), data format [56], composition interactions (e.g., querying and registering [51][190] [44]), composition operations (e.g., discovery [61]), behavior (e.g., traffic congestion was described using VSA [64]), functionality (e.g., actuation, communication, controllability, manageability, measurability, monitorability, physical context, sensing, states, and uncertainty [41]) or all that preceded (Using LNT to model a composite system's objects, interactions, states, and actions [55]). It's worth mentioning that formal modeling

TABLE 17. Interoperability challenges and solutions in IoT/CPS service composition.

Ref	Context	Interoperability Challenges	How Addressed
[18]	Smart End-to-end Massive IoT Interoperability, Connectivity and Security(SEMIoTICS)	- Technical: various device interfaces - Syntactic: various data formats or encoding - Semantic: various data models/ontologies - Organizational: heterogeneous APIs prevent communication between organizations.	• Standard device interfaces would address technical challenges. • Unified data formats/encoding would address syntactic challenges. • Unified data models/ontologies address the semantic concerns. • Standard APIs would solve organizational concerns.
[34]	DIY IoT Networks and Architectures	- Heterogeneous IoT systems and Platforms APIs.	• Interoperability in IoT across stakeholders and producers/consumers will only be achieved if standardized interfaces are provided.
[36]	Fair VS Full interoperability in ARM-enabled IoT Architectures	-IoT Systems with different Architectures.	• IoT ARM is a tool that allows fair interoperability by enabling bridges between systems that don't share the same architecture.
[38] [39] [40]	NIST CPS Framework enabled Platforms	- Heterogeneous components and systems. - Data Interoperability issues.	• External interoperability achieved by standardized APIs. • Providing a common language for describing composite CPS.
[121]	Gaiasense: Smart Farming (SF) IoT Platform.	- Lack of Interoperability APIs due to the cost of networks and sensors needed to allow communication between SF systems	• The "Data Interoperability Zone" and the "Information Management Adapter" are introduced to facilitate data interoperability between SF systems within an NGSIv2-FIWARE implementation.
[146]	VITAL-OS: Operating System for Smart Cities.	- Semantic (Data-Models) and organizational (Cross-domain) interoperability challenges	• VITAL leverages W3C SSN-compliant semantics (including JSON-LD, known for its flexibility and simplicity) to ensure interoperability across diverse IoT streams and domains.
[166]	IoT platform for composing analytics data.	-Unstructured data -Heterogeneous Semantics	• Semantic Interoperability and Data Homogenization techniques exploit both structured and unstructured data.
[173]	Fiware Composite IoT Applications	- Heterogeneity of communication protocols and data formats.	• Lightweight and reusable interoperability models that support a broad range of applications.
[202]	Interoperability and composability of IoT Web Services	-Traditional way of ensuring network interoperability (using standards) has shortcomings, including a lack of compatibility between some standards and specifications.	• A Technique that wraps service semantics into middleware at the application layer, which automatically builds APIs allowing interoperability without modifications to existing standards, devices, or technologies.
[204]	SenaaS: Event-driven IoT Sensor Virtualization/Cloud Approach	- Technical interoperability challenges related to the heterogeneity of IoT devices.	• A hard-coded adapter is used to mitigate the diversity issues related to sensor platforms (Technical) by selecting only compatible devices. Automated selection is planned for future work.
[255]	Composition of heterogeneous IoT services in smart homes.	- Heterogeneous devices come with different communication APIs and supported protocols.	• Using a unified communication protocol (UPnP) at the pervasive device layer to achieve communication between the heterogeneous smart home devices. • Once communication complexity is handled at the device layer, an AI planning technique is used to compose services during runtime.
[256]	Interoperability challenges when composing smart-city services in a multi-platform environment	- Lack of interoperability among IoT smart city platforms and services prevents IoT from reaching its full potential as computation load distribution is not managed efficiently.	• Analyzing the individual performance of a single IoT platform (normal workload, parallel services workload) and the aggregation of requirements when binding all platforms services within a smart city. • A Poisson Distribution function was provided to estimate load and ensure that one platform interface doesn't carry most of the load.

typically preserves compositions process type: for example, in [26], the concurrent/parallel nature of the composition is preserved by the model.

→ **Non-Functional Properties**: this includes QoS properties of a composite system, including cost and price of a composition [52], energy efficiency [68], or power inefficiency [13], safety [106] (analyzed using the CWB-NC tool [95]), privacy [84], security [92], scalability [46], interoperability [82], and system performance indicators such as response time [189].

b) Formal Verification: to perform model checking -for verifying formal properties or assessing the model against state space explosion-, the formal model needs to be loaded into a formal verification tool after translation to a formal specification. This is the case, for example, for PLUSCAL - converted to the TLA specification in the TLA+ tool- or LNT -a description language, which is later translated to LTS, a formal specification language- to formally verify the different properties of interest [50]. Formally verifiable properties mentioned in the primary studies discussed in Table 6 include correctness [60](verified in cloud environments using tools such as NuSMV[103]), compliance [63] , liveness (the overall study of formal verification problems such as starvation, deadlocks, and livelocks) [109], compatibility [108], non-link redundancy[58], privacy [115], security [84], safety [112], reachability [63] , livelocks [78], dependability [241], reliability (verified in PRISM [80]) [48][82], realizability

TABLE 18. Privacy challenges and solutions in IoT/CPS service composition.

Ref	Context	Privacy Challenges	How Addressed
[18]	Privacy aspects and dependencies in IoT and Industrial IoT Platforms	- Composing and facilitating the design of IoT platforms that are secure and privacy-aware.	● Privacy in IoT composition platforms is defined as a higher property and is decomposed into specific and low-level aspects each studied thoroughly, including authentication, authorization, data protection, unobservability, anonymity, unlinkability, undetectability, pseudonymity.
[29]	Privacy-aware IoT-A based FIWARE cloud IoT platform.	- Allowing trusted devices and objects to anonymously join and be composed within IoT-A compliant platforms (FIWARE).	● The Security Functional Group (FG) handles security and privacy issues in IoT-A-compliant IoT systems. ● The Identity Management Functional Component (FC) within the Security FG issues/manages pseudonyms/accessory information to trusted subjects, thereby ensuring anonymous operations and privacy.
[34]	Privacy best practices during the development of IoT Platforms.	- Developers focus more on functionality and less on ethical values related to the use of communication technologies. - Privacy policies may be in place, but their technical implementation is neither supervised nor adhered to.	● Human-Centric Computing is proposed as a solution for developing privacy-aware platforms from the architecture/development phase. ● Examples of Human-Centric Computing include the “Human Centric Systems Development Life Cycle”.
[36]	Privacy in ARM-based IoT Platforms	- Subject’s privacy is an "Element to Protect": data that a user or a device does not explicitly agree to make publicly available. It is specifically impacted when the composite system’s requirement includes non-repudiation.	● Privacy is addressed by leveraging an Identity Management component run by a trusted third party for user privacy protection and tracking malicious actions.
[38] [39]	Addressing Privacy in Cyber-Physical Systems using the NIST CPS Framework	- Certain types of CPS data may present little or no privacy concerns in isolation, but when aggregated with other data, they might become privacy intrusive.	● Privacy risk management guidelines include analyzing privacy risks throughout the entire data lifecycle: creation, collection, composition, exploitation, and disposal.
[115]	AI-driven composition and privacy validation in IoT	- Tangibly assessing Privacy is a challenge in IoT Platforms. - There is a need for IoT design and management frameworks that implement mechanisms to assess privacy.	● CompoSecReasoner addresses privacy concerns by deriving and validating privacy computation/estimation post-composition. ● Privacy was computed and estimated based on tangible vulnerability, exposure, and disclosure metrics. ● "Attack Surface" and "Medieval Castle" are security approaches used to derive measurements for privacy aspects based on "Attackable Points" and "Protection Levels."
[166]	Building IoT Analytics Platforms with privacy in mind	- Users need privacy policies that are permissive enough for services to work while also restrictive enough that their privacy is not compromised.	IoT Analytics platform builders must keep privacy in mind by: ● Anonymization of personal data. ● Encrypting and securing data storage ● Implementing user-customizable data sharing mechanisms.
[175]	Privacy-Preserving Smart-City IoT Platform based on ML and Blockchain	- SVM ML classifiers require labeled IoT data that may contain privacy-sensitive elements.	● secureSVM: a privacy-preserving SVM training scheme over blockchain-based encrypted IoT data. ● IoT data are encrypted and then recorded on a distributed ledger.
[176] [253]	Privacy-preserving in distributed and cloud IoT platforms using AI and Blockchain	- Data containing sensitive personal information can reveal the identity of IoT stakeholders, including consumers and producers.	● Human-centric approaches: user consent. ● Technological approaches: cryptography, blockchain networks. ● Regulatory approaches: GDPR. ● Blockchain provides a distributed way to protect privacy, as there is no centralized entity to manage the credentials of devices and stakeholders.
[219]	Consent and Trust related to Personal Data Sharing in IoT	- The IoT data volume doubles every two years, with more than 60% generated by individuals, with wearable medical devices data, particularly raising privacy concerns.	● The EU’s GPDR is proposed as a framework for addressing privacy by enforcing mechanisms of Transparency(how data are processed), consent(user’s ability to opt-in or opt-out), and erasure(right of the users to delete data).
[231] [232] [254]	Privacy-Aware Cloud or Cross-Cloud Service Composition.	- Private clouds don’t disclose QoS details of their service owing to business privacy concerns in cross-cloud scenarios. - Cross-clouds credibility might be doubted if the advertised service composition SLR doesn’t meet real specifications.	● The HIstory REcord-based Service Optimization MEthod (HireSomeII) protects cross-clouds privacy by evaluating their QoS history records rather than their advertised QoS values, which enhances the credibility of the composition plans they provide.

[86] , deadlock freeness [94], consistency [100] , non-conflict [58], conformance [56] , completeness [100], fairness [26], trustworthiness [41] , and communication properties such as unmatched send messages [21]. It is worth mentioning that verification of some of these properties can be automatically achieved when other properties are verified. This is the case in [98], where safety was verified through the non-reachability of deadlock states by leveraging the PROMELA

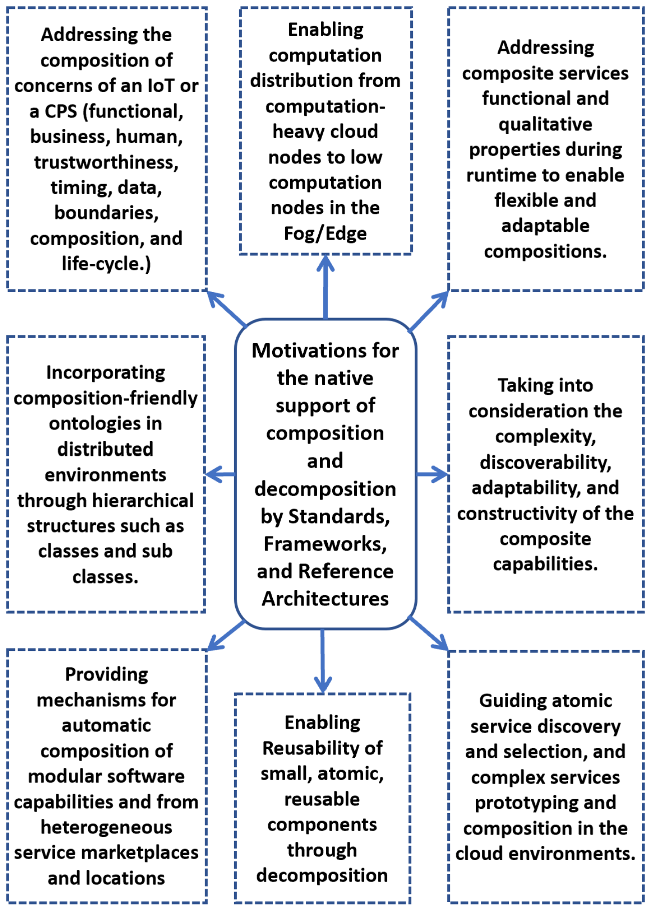
Motivations for the native support of composition and decomposition by Standards, Frameworks, and Reference Architectures
Addressing the composition of concerns of an IoT or a CPS (functional, business, human, trustworthiness, timing, data, boundaries, composition, and life-cycle.)
Enabling computation distribution from computation-heavy cloud nodes to low computation nodes in the Fog/Edge
Addressing composite services functional and qualitative properties during runtime to enable flexible and adaptable compositions.
Incorporating composition-friendly ontologies in distributed environments through hierarchical structures such as classes and sub classes.
Taking into consideration the complexity, discoverability, adaptability, and constructivity of the composite capabilities.
Providing mechanisms for automatic composition of modular software capabilities and from heterogeneous service marketplaces and locations
Enabling Reusability of small, atomic, reusable components through decomposition
Guiding atomic service discovery and selection, and complex services prototyping and composition in the cloud environments.

FIGURE 12. AQ1: motivations for the native support of service composition/decomposition mechanisms in standards, frameworks, and reference architectures.

formal specification.

Figure 13 presents the main properties of formal representations as learned from the primary studies results in Tables 5 and 6.

3) AQ3: What are the current trends in formal representations modeling and formal verification?

Five trends were pointed out from the data extracted in Tables 5 and 6. These trends are represented in Figure 14 as follows: formally verified functional properties (A), formal verification tools (B), formal verification techniques (C), formal representations modeling approaches (D), and formal representations - used in service composition- source code availability (E).

For formal verification techniques, model checking [53][81][102][104][105][107][113][116][236] represents the major technique for assessing properties of interest (74% of primary the studies in Table 6), while equivalence checking [87][114] and theorem proving [111][235] represent the remaining 26%.

70% of formal representations are algebraic, while graphical or hybrid representations representing are used in 30% of the identified primary studies: the use of algebraic solutions in most primary studies can be explained by the fact that they can be easily translated to formal specifications compared to their graphic counterparts, which require additional interpretations and transformations (e.g., in [86], BPMN 2.0 is converted to LOTOS NT, and the latter is transformed to the formal specification LTS using the CADP tool).

Verifying certain properties of composite systems is the main goal of formal verification. From this perspective, correctness remains the most verified property (%31 of primary studies). At the same time, safety, security, deadlock freeness, and reliability -combined- are addressed in 38% of the primary studies that addressed formal properties assessment. Properties such as privacy can be considered emerging properties in which researchers need to invest more effort (formally verified in one manuscript [115]), which also explains why it was studied in a specific RQ (RQ15). In terms of formal verification tools, CADP, TLA+, Maude, CoQ, and MWB [93] combined represented 55% of the tools leveraged in the primary studies, followed by Isabelle, UPAAL, CWB, NUSMV, and PRISM (5% each). These tools are used by both the research community and industry stakeholders.

From an implementation perspective, the data in Table 5 shows that 75% of the formal representations can be traced to a Git repository containing substantial examples of formally specified models for different composite systems. This provides resources to inspire researchers to model and verify the properties of novel services. Figure 14 highlights all these trends.

4) AQ4: What are the different ways formal verification techniques and tools tackled the state-space explosion problem in service composition?

Data extracted from primary studies in Table 7 show that the state space explosion problem in service composition can be tackled using three different approaches:

a) Model simplification:

This approach works better for models that cannot be decomposed into simpler problems. Examples of this approach include [188] (where a pi-calculus composition algorithm was simplified using an approach called unification that eliminated the state space explosion problem), [96] (where the state space was reduced by using Boolean functions instead of explicit representations with more states), [56] (where the FDR2 algorithm replaced the old FDR, yielding simpler models and as a result fewer states), [44] (where the algebra's order was reduced causing fewer states), [164] and [189] (where machine learning is used to determine the optimal service composition, which in turn simplifies the model), and [49] (the model was simplified by anticipating and eliminating errors in systems with a large state space).

b) Model decomposition:

This approach works better with models with a large state space that can be decomposed into simpler problems, making

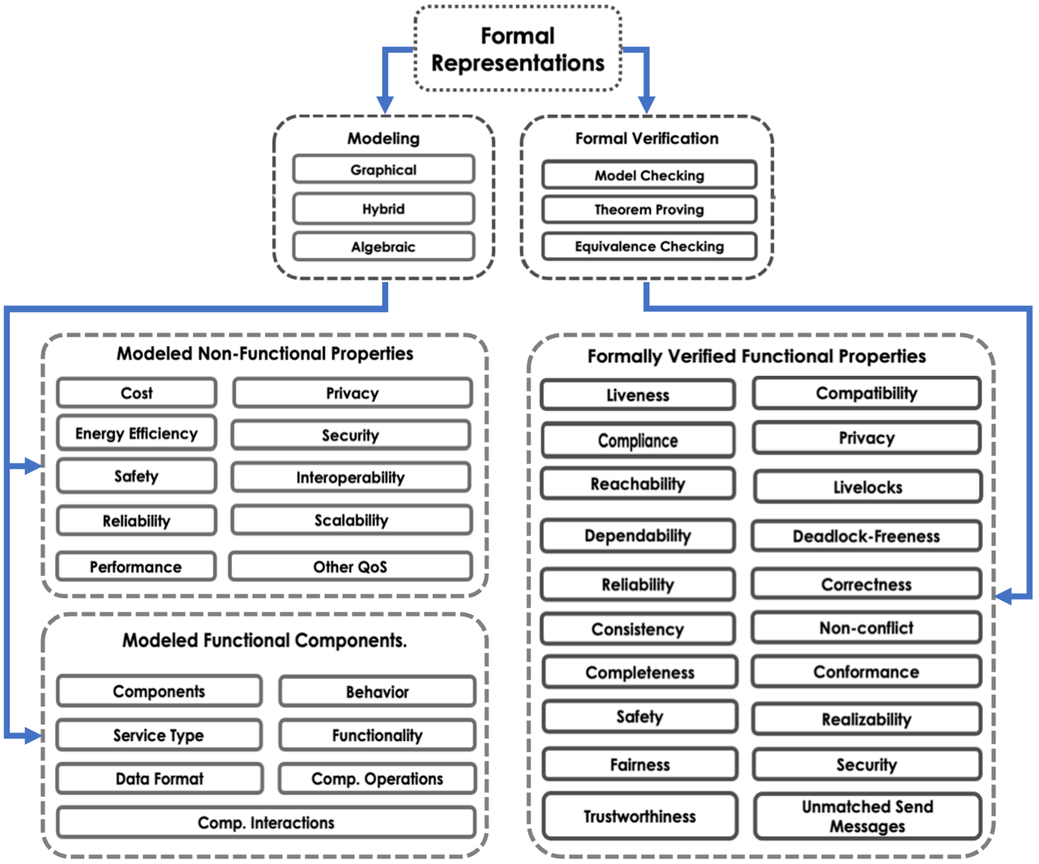

FIGURE 13. AQ2: Formal representations modeling and verification properties and techniques and corresponding modeled/verified properties in service composition.

model checking much simpler and preventing state space explosion. Examples of this approach include [54], where the CADP toolbox uses software mechanisms to decompose the model to be verified, also [58], where the size of the specification was reduced as a result of decomposing the model, and [238], where the model was decomposed into multiple sub-models, each with a smaller state space that can be checked individually, thereby reducing the state space and as a result preventing the state space explosion problem.

c) Resource management:

This approach leverages available hardware or cloud resources to alleviate the computation of formally specified models or to implement measures against eventual crashes when the state space computation exceeds the available resource capacity. Examples of this approach include [100], where the TLA specification makes use of multi-threading and allows access to multi-core processing or by offloading and distributing computation among multiple AWS EC2 cloud instances, or [190], where the simulator manages the state space explosion problem by stopping the simulated model to prevent crashes when the space is too large, which doesn't reduce or eliminate the state space explosion problem.

To conclude this analysis, the outcomes of these three approaches vary from reducing the state space explosion problem by making the model work for low-scale models, eliminating it if the model's state space is small enough to not cause the state space explosion, or managing it by preventing crashes in formal verification tools. The trends in Figure 15 (A) show that model simplification remains a widely adopted approach to tackle the state space explosion problem, and in 65% of the cases, the problem is reduced, as Figure 15 (B) shows.

5) AQ5: What are the different stakeholders' categories and concerns when composing or consuming composite capabilities in different domains?

Table 8 presents data from a set of primary studies that focused on different stakeholders' concerns regarding composing or consuming composite capabilities in different domains. To better understand these trends, we put up Figure 16, which matches pointed-out stakeholders and domains with corresponding concerns. Although Figure 16 doesn't represent a full picture of the pointed-out domains, stakeholders, and concerns, it highlights key relationships and synergies between these sub-aspects. Three main stakeholders were recognized: users, developers, and city planners (or

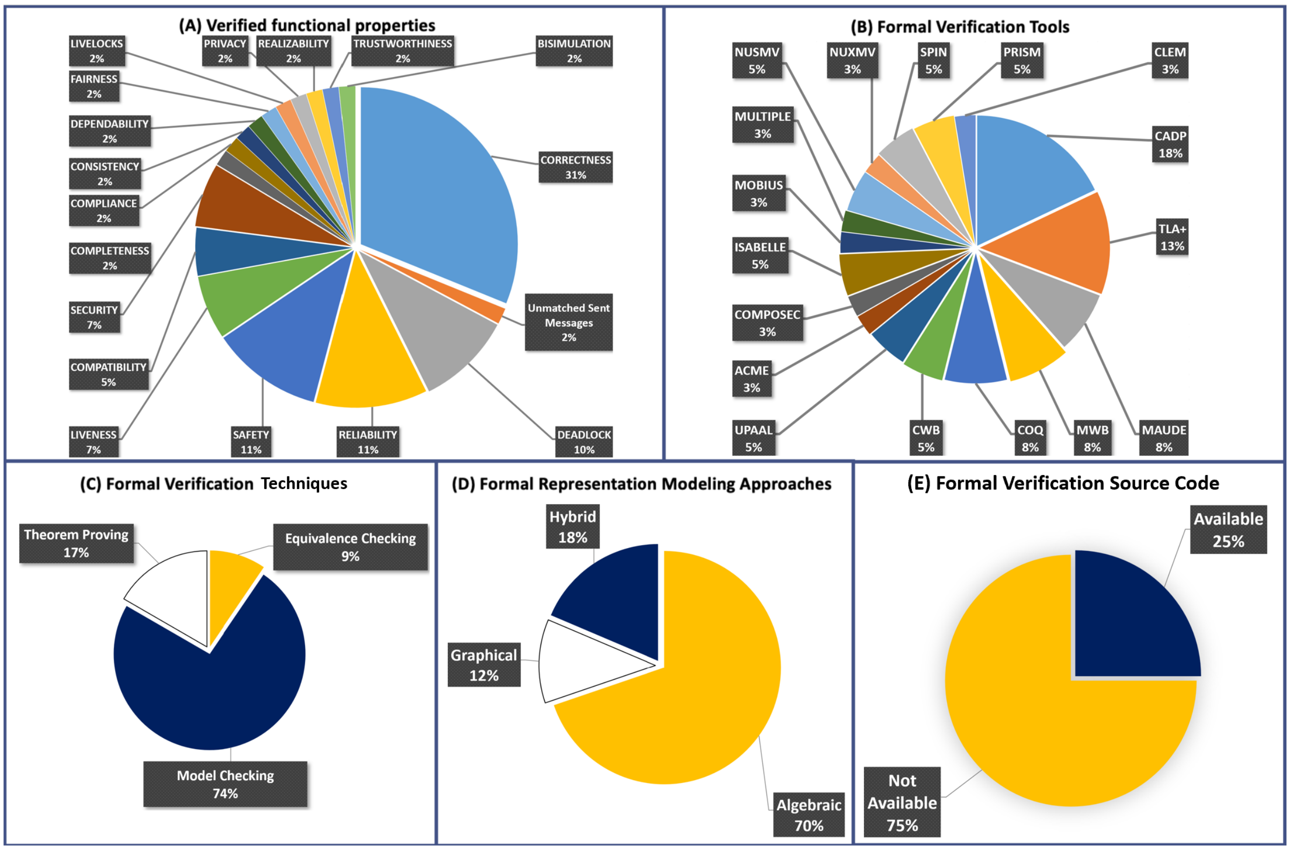

FIGURE 14. AQ3: formal representations: modeling and formal verification trends.

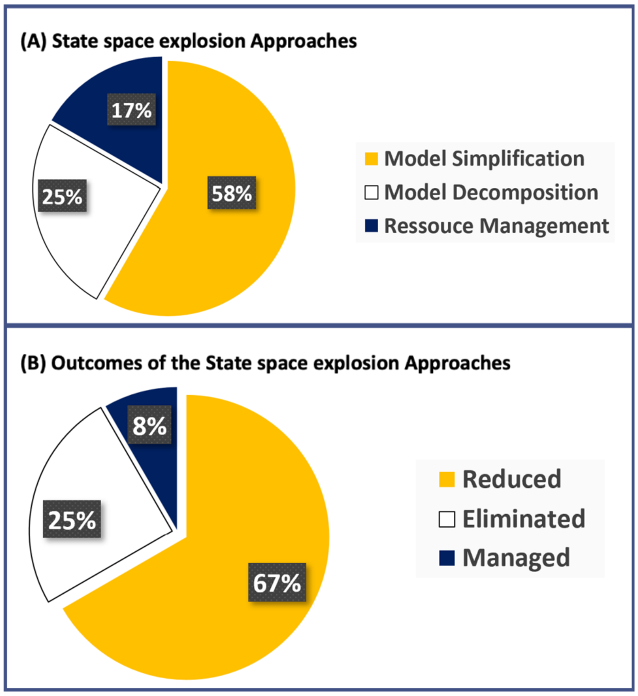

FIGURE 15. AQ4: trends related to the state space explosion problem approaches and outcomes.

platform managers), with the majority of efforts focusing on users (47%) and developers (42%) concerns.

For **Users**, the friendliness of composition platforms from a GUI perspective represents an important concern, this is the case for smart buildings applications such as [117], and this has been mentioned in other domains, including smart transportation [197][199]. Customization and ease of use are also user concerns in the smart manufacturing domain [203]: customers use recommendation algorithms to customize products during pre-production in terms of cost, reliability, and delivery time, among other criteria, [181][125]. Users also expect energy-efficient composite systems in smart cities [165][178][179][180], less human interaction and more automation in composition platforms processes [23], ease of use of composition platforms[26][27], cost reduction or zero-cost implementation [121], trustworthiness concerns that include the human/user factor in smart manufacturing [41], accessible safety assessment especially for critical metrics such as security in smart cities [216], safety in smart transportation applications, data privacy in smart health applications [119], environment friendliness and security [216] for smart buildings and cities.

For **Developers**, most concerns relate to the ease and time to develop, customize, prototype, and publish composite services. This typically requires ready-to-use APIs or at least available and less complex software objects for hardware and communication components. Ease of development, customization, prototyping, and publishing of services for developers were pointed out as concerns in applications related to the domains of smart health [119], environment monitoring [51], and smart cities [20][120]. Other developers' concerns that are worth mentioning are collaboration and interoperability in smart manufacturing [124], smart transportation [64], and environment monitoring domains [51]. These concerns contribute to a faster and easier composite service development process.

For **City Planners and Platforms Managers**, four concerns were identified: customization of services to clients, which is the case for insurance companies that can use composition platforms such as Dracena to generate costumer-specific fine-grained recommendations on insurance fees [16]. Another concern is cost reduction. An example of such concern is addressed in [120], where city planners encouraged developers to adopt modular and reusable designs for composite services for cost optimization purposes. An additional identified concern is ensuring trustworthiness in smart manufacturing. An example of this concern was mentioned in [41], where researchers proposed a cloud manufacturing platform that considers trustworthiness pillars as indicated by standards such as the NIST CPS Framework. Finally, energy efficiency in smart cities was recognized as a major concern by both users and platform managers in [165] [178][179][180].

6) AQ6: What are the technical differences in capabilities composition implementation in different platforms ?

Primary studies pointed out in Table 9 were selected to answer this question as they provide sufficient implementation details, including where the services are implemented from an IoT layer perspective, which composition engine is used, and implementation instructions that would highlight more details about the technical differences in implementation.

From a platform nature perspective, four implementation trends were recognized:

a) Cloud/fog implementations:

The implementation of these platforms leverages cloud services, communications, and capabilities to provide value-added services. Examples of such implementations include MCC Cloudlets [30], where researchers have created virtual objects representing edge devices at the fog level. These virtual objects were later composed into applications in the Central Cloud. Networking between the edge, fog, and cloud components is ensured via OpenStack, and the repository for virtual objects is maintained at the fog level. Thing Orchestration [191], along with Cloud CAMP [132] are two other examples of composition performed at the cloud level. Another example of the use of cloud computing capabilities to compose novel services or applications can be found in both commercial solutions: AWS GreenGrass [126], and Microsoft Azure IoT [129] [128]. AWS GreenGrass leverages Lambda scripts to compose value-added services and applications by connecting different AWS services; this requires the installation of the different GreenGrass dependencies and certificates related to the target devices and services. Similarly, Microsoft Azure IoT relies on the Azure IoT Hub cloud component to connect and compose capabilities and services at the cloud level. This also requires the installation of different Microsoft Azure components and command line capabilities. A rule-based cloud solution, IFTTT [37], has also been presented as a cloud composition service. Users can set sensing or actuation rules anywhere on their apps, and the IFTTT service composes the desired rules or outcomes. Finally, Fiware [29] is also a cloud composition platform that leverages multiple plugins, including the Entity Composer Plugin, to compose capabilities provided by devices or services at the edge or fog level, which requires special agents to be installed on edge devices or services to allow the composition of capabilities at the cloud level.

b) Edge/Local containers implementations:

Edge implementations rely on edge devices to perform a subset of compositions and computations when these edge devices satisfy minimum computation requirements. Edge computing or composition can also be leveraged to minimize delays in safety-critical applications. This is the case for [242], where mobile edge servers are proposed as a solution for composing capabilities at the edge when the edge servers have sufficient computational power. When edge servers are overwhelmed, only then do cloud center nodes take over the composition computation. This type of composition is typically suited for delay-sensitive applications such as smart transportation applications. Another example of the use of edge computing to perform service composition is highlighted in [51], where authors discussed Home Assistant: free and open-source software for home automation designed for central control in smart homes. Home Assistant is hosted on a local machine and doesn't require internet connectivity. Edge devices and the Home Assistant node are located in the home LAN and are managed in a way that ensures privacy and security. mPlane is another measurement platform that can be used for network measurements, and compositions at the edge [12] and requires a local node running the composition engine (Supervisor) on a competent machine connected via different types of APIs (TCP/IP, REST) to network probes, services, and clients. A similar concept is highlighted in [140], where the Vert.X toolkit is used to create and compose local microservices, with the ability to call remote atomic capabilities via different APIs such as Axios. Finally, the publication [13] highlights an example of edge/local composition using a composite engine (Fast IoT Composer) running on a local machine to compose smart home services.

It can be concluded that edge/container composition only serves as a substitute for cloud composition when the resources associated with edge/container nodes are sufficient to perform the composition computation. From a performance

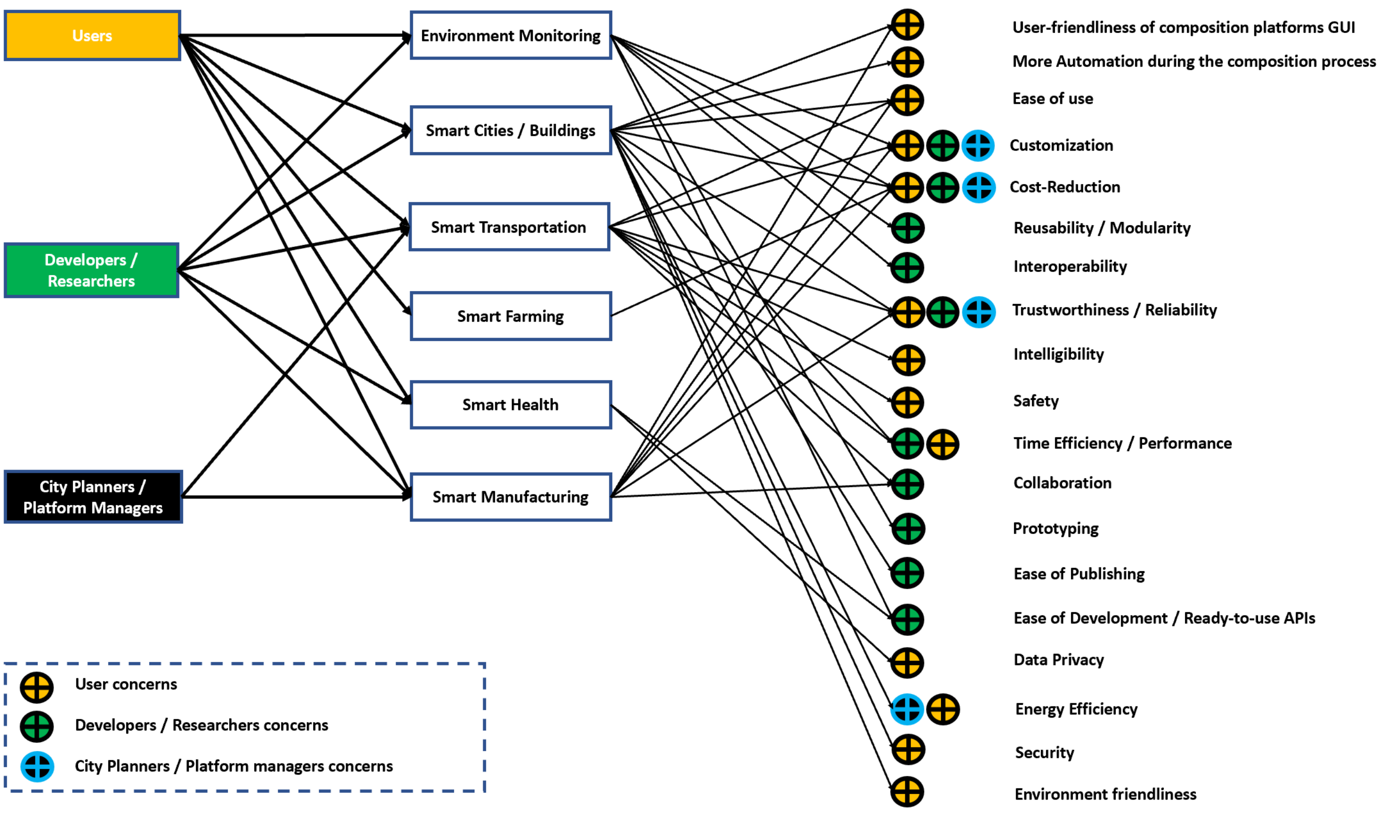

FIGURE 16. AQ5: primary studies trends related to stakeholders' concerns in different domains.

perspective, edge composition is faster and provides faster results and services than its cloud counterparts. Finally, it is also safe to assume that edge/local containers computing provides more privacy when the data doesn't leave the user's vicinity.

c) Simulation/prototyping implementations:

These platforms compose simulations or prototypes for composite services and systems. Examples of these platforms include [136] [118], where a CPS representing an autonomous vehicle for safety assessment was simulated in UCEF using atomic components called federates that communicate using the pub-sub HLA protocol. Another example can be found in [64], where services and sensors can be fetched using REST APIs from Node-RED's web interface, which enables the simulation and prototyping of composite services. It is worth mentioning that Node-RED can be used for production as well for less-complex home applications, but other mature solutions are well-optimized for such purposes, such as Home Assistant.
Trends in Figure 17 show that from a platform nature perspective, cloud composition platforms represent (43%) of the primary studies identified: this can be explained by the fact that most studied compositions are computation-intensive and need resources that typically reside in the cloud. In addition, cloud platforms are mostly commercial with high maturity and reach as opposed to edge/container solutions which are mostly academic efforts. Edge or local containers

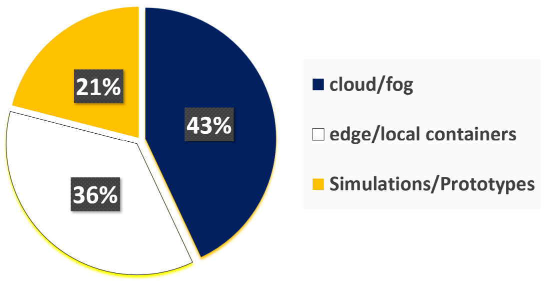

FIGURE 17. AQ6: implementation trends in service composition.

composition solutions represent (36%) of primary studies, whereas service composition simulation platforms represent only (21%) of primary studies.

7) AQ7: What are the most common composition processes, and how do they differ in terms of automation level?

The primary studies in Table 10 were selected as they provided input on the composition level and the composition process type. By analyzing Table 10 columns, especially the automation and process type manifestation columns, the depth of complexity and automation in each process type can be learned. Two trends were pointed out:
a) Low complexity processes => Require no/basic user input: These processes facilitate the automation of composition owing to their low complexity, and they are ready to exploit APIs and interfaces to atomic services [26][27] or by automating complex composition steps, including data collection, and composition [17][64][46][32][33]. Rule-based

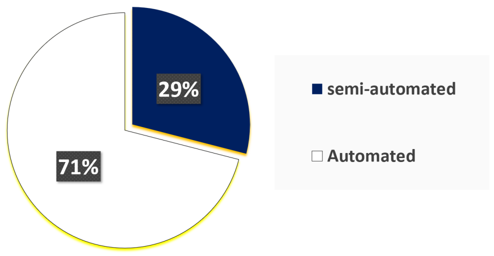

FIGURE 18. AQ7: trends in service composition automation level.

processes can be considered the least complex of all composition process types because they only require setting up simple rules or composition goals by the user from a GUI to trigger desirable events [147][148][21].

b) High complexity processes => Require user or developer input and programming These processes have a higher level of complexity since they require extra programming to achieve the composition goals. Examples of such composition process types include process or programming-based composition processes, which -as the name suggests- require manual programming or crafting composition scripts to achieve the compositions goal [51]. Asynchronous or parallel compositions also show a higher complexity level than their synchronous counterparts, which was indicated in [16] and [12] where additional development on specific highly programmable platforms is required. Finally, some flow-based composition processes can have a high automation level but still require user input on specific QoS parameters to successfully achieve the goal of the composition, as in [13], where user input is required during the service selection step to recognize specific atomic services that meet certain QoS properties of interest.

Figure 18 aggregates these trends and shows that among the 14 primary studies addressed in Table 10, 28% are semi-automatic for their higher complexity and their need for user or developer input. In comparison, 71% of the primary studies were automatic for their lower complexity and their need for basic or no user input to achieve the composition goals.

8) AQ8: What roles do communication protocols play in composing or decomposing IoT capabilities ?

Table 11 contains the primary studies that provide insights into the role of communication networks in service composition or decomposition. Three categories of roles were pointed out.

a) Communication roles: This role is manifested in i) enabling a wide variety of services such as microservices (Vert.X's Axios, [140]) and pub/sub Java federates (HLA [136],[118]) to communicate and exchange data with composition engines to build composite services, or in [211], where smart connected IoT devices leverage the mashup paradigm to communicate and exchange data, leading to value-added services through compositions in WSNs. ii) facilitating the communication of heterogeneous IoT networks through transparent access (CoAP/HTTP, [160]); iii) enabling the discoverability of IoT devices capabilities (mDNS, [21]); iv) simplifying interfaces and enabling flexible network management (SDN, [27][41]); v) acting as middleware between publish-subscribe brokers and services (MQTT,[122][147]); and vi) offering commands that enable requesting, receiving, composing, and storing data by leveraging multiple network components (mPlane protocol, [12]).

b) QoS-related roles: This role is observed in (CoAP, [24]) which allows communication between energy and computation-constrained devices, or in (SVOM, a CoAP-based protocol, [24]) which captures the status of registered devices to achieve fault management. In [210], energy efficiency was enabled in SOA composite applications in WSNs by leveraging the NOC paradigm. Similarly, in [169], an EPCglobal-compliant middleware was proposed to facilitate communication between IoT devices in a way that ensures scalability, elasticity, service decomposition capabilities, multi-threading, and cloud virtualization.

c) Process roles: In addition to communication and QoS roles, two process roles were identified in primary studies that communication protocols facilitate in service composition: i) enabling asynchronous communications (REST, [22]), and ii) enabling service decomposability by tracing back interests requests to reveal which capabilities (atomic or composite) exchange data with the composition engine or clients (NDN [161][162]).

9) AQ9: What are the main roles of data models leveraged in service composition and decomposition ?

Primary studies used in the discussion below were selected because they provide the reasons behind using a given data model for composing capabilities. These primary studies also provided concrete examples of such data models, as shown in the last column of Table 12, where a subset of the data models used and their attributes are represented. Some attributes are shared by most data models, including capabilities ID, location, and timestamp. Based on the data provided in Table 12, the following data models and schemas roles were pointed out:

a) Modeling services, composition, decomposition, processes, and operations: The ease of composing atomic capabilities is one of the key enablers for creating value-added applications, which can only be done through the use of data models that facilitate this task by representing capabilities data and operations. This is highlighted in [25], where JSON is used to model mPlane capabilities thanks to its simplicity, parseability, and efficiency. JSON is also used to represent services, bindings between services, as well as the strength of these bindings [26][27][192]. In IFTTT [21], JSON Web of Things (JWT) is used to model objects and composition rules. The same applies to WSDL [44][209], which is typically used with SOAP and XML [201][206] schemas to describe and enable the discovery of web-services. BPEL data models describe composite services processes and enable their deployment in three steps: process template creation, process composition, and process installation. Another flavor of BPEL, WS-BPEL [156][205], describes services and their execution and compositions [153], with BPEL-TC upgrading the features of WS-BPEL to include composition and decomposition requirements for temporally customized web services [155]. OWL-S enables nested classes description to model functional and non-functional properties of IoT devices [183][157], and OWLS-TC4 provides simple capability aggregation mechanisms for non-complex compositions [123]. Finally, HSML/HTML [51] describes the state of IoT services whereas SSN [158] represents context information for capabilities including deployment attributes. The attributes of data models illustrated in Table 12 show synergies and common attributes (id or name, timestamp or start time, location or ip, etc.) between most of the different data models when it comes to describing services.

b) Enabling formal specification: After adequate translation, some data models can enable formal specification. The translation process complexity varies in terms of difficulty from one data model to another. Examples of data models converted into a formal specification include JSON [64], which was used for a service description of the Node-RED object detector, which was later converted into semantically comparable service vector descriptions (VSA: Vector Symbolic Architecture). Another example of this role can is found in OWL-S [110], which enables services description (service profile, service processes) based on which a TLA formal specification is defined. Some efforts mentioned challenges that complicate the conversion of data models to formal specification: an example of such complications can be found in WSDL [44], which cannot be converted into formal specification because of its lack of implementation details and logic.

c) Facilitating interoperability between different platforms through integration APIs: To allow different platforms to compose their capabilities when their data models differ, an interoperability bridge between these platforms must be established. Data model integration APIs are one of the bridge techniques used in service composition to accommodate this requirement. Examples of such techniques can be found in [152], where JSON-LD was easily integrated with other Context Information Management (CIM) platforms, and RDF/XML (Resource Description Framework) was used to integrate APIs with external XML platforms to ensure backward compatibility. XML has also been used in mPlane [25] to allow integration with external platforms using XML data models.

d) Documenting services and debugging processes and compositions: YAML's improved human readability and writability, making it suitable for documenting and debugging [25].

e) Special roles: WoTDL [149] is an alternative to existing WoT models such as OWL-S and WSDL, which are unsuitable for describing AI planning concepts for automatic WoT compositions.

10) AQ10: How are atomic or composite capabilities quantified or measured?

Data from the primary studies identified in Table 13 is used to shed light on the measurability aspect of service composition. Regarding atomic or composite capabilities, the following measurability trends were recognized:

a) Atomic capabilities:
↪ The are typically concrete and tangible capabilities with standardized metrics and units.
↪ They are typically generated by devices by converting physical environment information to a digital form and associating it with a standardized unit.
↪ Cannot be further decomposed into other atomic capabilities.

The best example of atomic capabilities measurability approaches that satisfy the trends above is illustrated in [15], where functional atomic capabilities (temperature, humidity, carbon dioxide, wind speed) and non-functional atomic capabilities (RTT, latency, and drop rate) of a thermal comfort system were measured using different weather and home sensors as well as system performance probes.

b) Composite capabilities:
↪ Are typically abstract/non-tangible capabilities; the metrics and units used are user-defined and non-standardized.
↪ Are generated by combining and aggregating atomic capabilities in a way that end users can assess.
↪ Can be decomposed into atomic capabilities.

Examples of composite capabilities measurability approaches that satisfy the elements above include the two non-functional capabilities mentioned in [26][27]: i) "Tool Usability": was assessed using many atomic factors derived from the user's experience including the number of clicks to achieve a composition task and the easiness of interacting with the UI. ii) "Platform Performance": was assessed by assessing and aggregating multiple factors, including the time for the deployment and the time to generate a specification, as well as the number of states and bindings processed by the platform. Examples of functional composite capabilities include Traffic Jam Trend prediction [16] (composed based on speed and location atomic capabilities provided by the SUMO simulator probes) and Thermal Comfort [15] (composed based on atomic capabilities provided by weather sensors: temperature, humidity, carbon dioxide, and wind speed). The functional and non-functional composite capabilities identified in Table 13 follow the trends mentioned above.

It can be concluded that measurability trends apply regardless of whether atomic or composite capabilities are **i) functional** (inherent features of IoT devices and represent their main output. For example, a temperature sensor provides a functional capability: measuring temperature, which is a measurement that can be expressed with a quantitative value and a unit (Celsius or Fahrenheit)) or **ii) non-functional** (QoS and performance metrics associated with the IoT devices -or the infrastructure they run on top of- rather than their main feature itself, these non-functional properties include the SLA level, latency, redundancy, scalability, interoperability, security, and privacy, among other non-functional features [167]). As for the composite IoT capabilities network infrastructure, measured aspects include service quality and network capacity [198][200] by tracking resource utilization. Other non-functional measurements in web-services platforms include performance, resource utilization, dependability, fidelity, response time, availability, and cost [212][215]. Finally, an interesting example of measuring latency in mPlane is worth mentioning as latency measurement in this example is a functional property as it represents the main feature of mPlane as a network performance measurement platform [12]. This same metric (latency) is considered a non-functional property in other platforms, such as the thermal comfort measurement platform [15], as it doesn't contribute to the calculation of comfort but rather provides an idea of the platform's performance.

11) AQ11: What are the benefits of building IoT platforms and complex services with decomposition in mind ?

The primary studies in Table 14 showcase publications focused on service decomposition. This is a particularly interesting topic that has not been investigated in previous SLRs. Based on collected data, the following main benefits were pointed out:

a) Capabilities reusability: Platforms with decomposition support benefit from this property as a decomposed complex service into atomic capabilities can see its atomic services reused in other compositions, omitting the need for implementing redundant services and saving relative costs. This is the case in a hierarchical IoT platform [170] where complex capabilities are decomposed and reused to complete the pieces of another service with higher complexity. Another example of reusing capabilities owing to built-in decomposition mechanisms can be found in [119]: a microservices-based platform allows larger services to be decomposed into small, focused, self-contained services with loose coupling, which facilitate their reuse. The same case applies to using Domain-Driven-Designs (DDD) to decompose monolithic software [171], which makes it possible to use pieces of monolithic software in other compositions. Finally, in [51], the FSM model-driven services decomposition broke their linkage, which enabled their reusability.

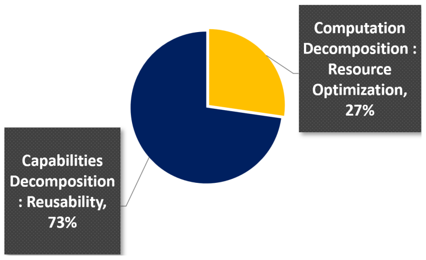

FIGURE 19. AQ11: benefits of building platforms with decomposition in mind.

b) Resource optimization: The decomposition of complex capabilities leads to resources optimization not only in terms of costs associated with deploying novel services [168] but also in terms of reducing network congestion as monolithic services consume more bandwidth compared to atomic capabilities[31]. The same concept applies to [169], where computation-intensive virtualized services leverage decomposition to optimize computation. Another case of resource optimization was encountered in [13] where decomposition leads to the identifying energy-efficient atomic capabilities that can be later used to compose efficient complex services.

In addition to capabilities reusability and resource optimization (main benefits), other benefits of building composition platforms with decomposition in mind were mentioned, including **stakeholders acceptance** as described in [168] where platform users can benefit from the reduced cost of exploitation due to reusability and resource optimization. Other benefits include **improved QoS parameters** such as platforms agility, flexibility, and scalability [166]. **Improving collaboration** between capabilities has also been pointed out as a benefit of decomposition in [64]. Finally, the **traceability** of the atomic capabilities that contribute to a certain composite service was mentioned as a benefit in Information-Centric Networks (ICN) such as Named Data Networks (NDN) [161] [162].

Figure 19 shows the distribution of the main benefits of building composition platforms with decomposition in mind as extracted from Table 14. Efforts that leveraged decomposition to break complex services into atomic capabilities for reusability purposes represent 72% of primary studies. In contrast, efforts focused on computation decomposition for resource optimization represent the remaining 28%.

12) AQ12: What role can AI/ML techniques play in shaping or improving service composition?.

The role of Artificial Intelligence (AI) and Machine Learning (ML) in service composition was not surveyed in previous service composition SLRs. We presented primary studies that tackled this particular aspect in Table 15; these primary studies show the roles AI/ML techniques play in service composition, including:

a) Composing capabilities with AI/ML features.

Primary studies in Table 15 highlight this role. In [251], AI/ML capabilities were implemented in a smart transportation platform that analyzed incoming vehicle data to build real-time automated driving features. In [20], complex atomic capabilities with AI/ML features were built and made available to developers to compose smarter systems and platforms in smart cities or smart transportation domains. Examples of these atomic capabilities with AI/ML features include multiple predictors such as noise level and free parking area predictors. Another example of the role of AI/ML in building complex capabilities was referred to in [31], where a cloud facial recognition composite service leveraged AI/ML atomic capabilities at the fog level, including facial features extraction, data fusion, data filtering, and face detection algorithms. Finally, in [120], an example of reusable AI/ML smart city services was studied in the context of a collaborative platform. This platform enables the composition of smart city services based on atomic AI capabilities, such as traffic and parking estimators.

b) Improving composition platforms and processes.
This is the most common role among the pointed out AI/ML primary studies aggregated in Table 15, which shows how AI/ML can play a role in improving composition platforms as in [248], where AI/ML is leveraged to improve maintenance of IoT composition platforms and in [249] where AI/ML techniques were used to improve security. To improve the composition processes, in [32] and [33], an AI-based classification method applied to a pipeline consisting of ML services was used to maximize the classification accuracy, leading to an improved service selection process. Similarly, a service composition technique called AI Planning was used in [163] for automatically composing well-described web services into feasible workflows and selecting and organizing web services based on location. [159] is another primary study where genetic algorithms were used in SOA-based environments to perform service selection based on QoS properties defined by the user. Finally, another example of AI/ML-based service selection was presented in [172], where a reinforcement learning agent was used in the dynamic of Mobile IoT to perform service selection based on user-defined criteria (including spatial cohesiveness, number of handovers).

A secondary role of AI techniques in service composition was discussed in [115], the CompoSecReasoner framework leveraged AI algorithms to assess, monitor, and verify properties such as security, privacy, and dependability. The components of the CompoSecReasoner framework also ensure dynamic system composition verification, property validation, and metrics-based automated administration.

Based on the primary studies in Table 15, Figure 20 shows the distribution of the main roles of AI/ML techniques in service composition: i) improving composition processes (67%) and ii) composing AI/ML enriched or improved capabilities (33%).

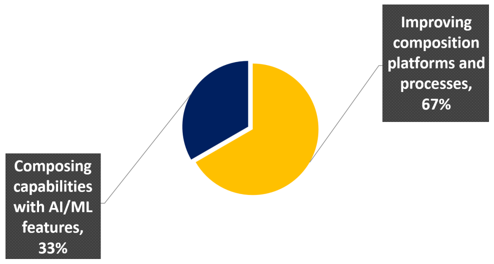

FIGURE 20. AQ12: identified roles of AI/ML in service composition.

13) AQ13: What are the main scalability challenges and solutions adopted when composing IoT and CPS capabilities?.

As novel services with high value require the composition of multiple atomic capabilities, scalability can arise as a challenge if the platform fails to scale for performance, QoS, network, or other constraints. Based on data from Table 16, the **growing number of services, users, and providers, generating or requesting composed and value-added services** represent the main causes for scalability issues. These challenges and potential solutions are discussed to provide answers to RQ14.

• **a) Scalability Challenges in service composition**
In [166], IoT BigData analytics platform scalability was impacted by a large number of devices, services, and users. Similarly, a large number of wearable smart health devices cause scalability issues not only from a computation perspective but also from a budget perspective [250]. For [175], the training data originating from multiple providers overwhelmed the training and composition algorithms in the secureSVM platform. Similarly, large user requests are considered the root of overwhelming a WS-BPEL-based online travel recommender [154]. Another example of the scalability issues caused by high numbers of IoT objects was mentioned in [26][27], which not only impacted IoT smart home applications deployments but also generated a large state space that caused the state space explosion problem when formally verifying composite capabilities properties. In [207], composition algorithms that did not scale for reasons associated with large user quantitative constraints were the subject of analysis. Finally, ultra-large numbers of services were also identified as the root of scalability issues in [45][46][47].

• **b) Scalability Solutions in service composition**
To address scalability challenges in the above primary studies, researchers have adopted the following solutions:

i) Properly adjusting and dimensioning resources: An example of such a solution was adopted in [166], where researchers proposed a dynamic solution for dimensioning resources, including storage and processing, to anticipate scalability issues. Another example of such an approach was pointed out in [154], where researchers proposed redundancy and load balancing at the computation and network resources level to tackle higher loads and user requests for composite services.

ii) Adopting optimized and enhanced composition algorithms to overcome scalability issues: This is the case in [207], where CP-net algorithms were proven to have a better capacity to handle compositions that require a large number of user constraints. Similarly, an optimized Gradient Descent Optimization algorithm was used in [175] for training and composing data, creating a platform that scales with a large number of data providers. Another example was identified in [26][27], where CADP's parallel algorithms and composition capabilities were leveraged to overcome scalability issues when formally verifying properties of interest.

iii) IoT capabilities layers, data, computation, and workflows management: Adopting a layered architecture that distributes computation across different layers (cloud, fog, edge) was adopted as a solution in [250] to handle the large amount of data generated by smart health wearable devices. The research around the DX-MAN composition platform also provides an example of this solution [45][46][47] where researchers highlighted scalability requirements that, when applied to data and workflows associated with IoT devices, contribute to better scalability, these requirements are: explicit control flow, distributing workflows among multiple compute nodes, localization transparency, decentralized data flow, separation of control, data, and computation, and workflows variability.

iv) Building IoT composition platforms based on scalability-aware frameworks: frameworks with scalability recommendations and guidelines can help researchers, developers, and composition platforms providers build platforms that scale from the get-go. An example of this approach is illustrated in the NIST CPS Framework [38][39][40], where building IoT/CPS capabilities composition platforms with the NIST CPS Framework guidelines in mind -especially those related to trustworthiness- would provide elements to enhance scalability in terms of network constraints and in the case of large numbers of users.

14) AQ14: What are interoperability challenges and solutions when composing capabilities from heterogeneous environments?

Capabilities in IoT or CPS may originate from heterogeneous devices and services. This can create a challenge when attempting to compose these capabilities to innovate value-added capabilities. Primary studies in Table 17 discussed both challenges and solutions to interoperability. These aspects are presented below to provide answers to RQ14.

• a) Interoperability challenges in service composition

Two main challenges to service composition interoperability were identified:

i) Heterogeneous services, systems, networks, and components: lead to a lack of communication/network interoperability. This was recognized as a challenge in multi-platform environments discussed in [256], where the lack of interoperability impacts load distribution. The same issue was noticed in the NIST CPS Framework enabled platforms when it comes to heterogeneous components, and systems [38][39][40], or IoT Systems with different architectures [36]. Similarly, researchers in [18] and [121] mentioned two challenges, one technical, related to the various device interfaces in heterogeneous IoT systems and platforms APIs, and the second organizational, illustrated by the heterogeneous service APIs that prevent communication between organizations. Another case of this challenge was identified in [173][255] related to the heterogeneity of communication protocols and the challenges it represents to interoperability, and in [34] where heterogeneous APIs of certain IoT systems and platforms contribute to the lack of interoperability and as a result constitutes challenges to service composition.

ii) Heterogeneous and unstructured semantics and data: Unstructured and heterogeneous data and data models were exposed as a challenge in the NIST CPS Framework [38][39][40], and in [166] as a challenge to easily compose useful analytics in IoT platforms. Similarly, the researchers in [18] and [146] discussed syntactic and semantic challenges, including various data formats, data encoding, data models, and ontologies as a hindrance to composing cross-domain IoT services. The lack of compatibility between data formats and data models standards and specifications also falls under the same umbrella as discussed in [202] and [173].

• b) Interoperability solutions in service composition

Two leading solutions were identified to address the interoperability challenges mentioned above .

i) Standardized or custom communication APIs and suitable interfaces were identified as a solution for bridging interoperability gaps and challenges in [173] [18] and would solve related technical and organizational concerns. In [256], understanding the load each platform interface handles in a multi-platform environment and providing suitable interfaces that fairly distribute this load is key to improving interoperability between the different platforms. In [34], providing standardized interfaces was pointed out as a measure to achieve interoperability in IoT across stakeholders and producers/consumers. Another similar solution was identity in ARM-based IoT platforms [36], by using the IoT ARM tool to allow fair interoperability by enabling bridges between systems that don't share the same architecture. [121] is another effort that adopted the same strategy; the "Information Management Adapter" was introduced to facilitate interoperability between smart farming systems within an NGSIv2-FIWARE implementation, and in [204] a hard-coded adapter was used to mitigate the diversity issues related to sensor platforms by picking compatible devices. Finally, standards

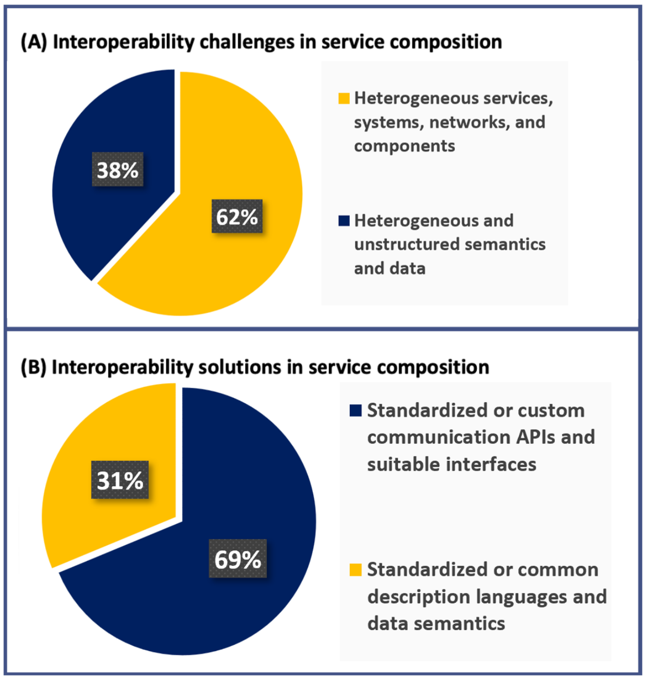

FIGURE 21. AQ14: interoperability challenges and solutions in service composition.

such as the NIST CPS Framework [38][39][40] proposed standardized APIs to achieve external interoperability and, as a result, facilitate service composition in cyber-physical systems.

ii) Standardized or common description languages and data semantics: is a technique adopted in multiple efforts such as [163] to ensure capabilities are composable by leveraging data homogenization techniques to exploit both structured and unstructured IoT devices data in analytics. In [173], lightweight and reusable interoperability models were used to support the composition of a broad range of applications, and in [18], unified data formats/encoding were leveraged to address syntactic and semantic challenges to facilitate service composition. Similarly, researchers in [146] leveraged W3C SSN-compliant semantics (JSON-LD) in the VITAL platform to ensure interoperability across diverse IoT streams and domains, and in [121] a “Data Interoperability Zone” was introduced to ensure data interoperability between smart farming systems within an NGSIv2-FIWARE implementation. In [202], a technique that wraps service semantics into middleware at the application layer automatically generates APIs allowing interoperability without modifications to existing standards, devices, or technologies. Finally, The NIST CPS framework [38][39][40] proposed providing common languages for describing services to ensure an easy composition of CPS capabilities. Figure 21 shows the above trends related to interoperability challenges and solutions in service composition.

15) AQ15: What are the main privacy challenges and solutions in service composition?

Primary studies in Table 18 focused on privacy challenges and solutions in service composition.

- **a) Privacy challenges in service composition**

Four categories of privacy challenges were identified:

i) Non-trusted objects/Devices Challenges: non-trusted objects anonymously joining composition platforms can lead to privacy issues when collecting or processing IoT Data. An example of this challenge was highlighted in IoT-A-based composition platforms such as Fiware [29].

ii) Data Challenges: data in IoT or CPS composition platforms can reveal private information when labeled, exchanged, or composed. In [38][39], the NIST CPS framework highlighted the privacy threat that originates from composing or aggregating certain types of CPS data, which may present little or no privacy concerns in isolation. Another example of data challenges leading to privacy concerns is found in [175], where the use of labels on data to allow Machine learning classification might compromise privacy if those labels contain privacy-sensitive elements. In [176], the ever-growing number of IoT devices generating privacy-sensitive information is considered a privacy concern if the data are not properly processed throughout its life cycle: a Special case of this issue is discussed in [219] where the ever-growing number of medical IoT devices constitutes a privacy sensitive challenge if the data are not properly handled.

iii) Platforms Design challenges: platforms that lack privacy components by design are the most vulnerable to different privacy challenges as discussed in [18]. In other instances, Privacy policies might be in place. Still, their technical implementation is neither supervised nor adhered to by developers, who typically focus more on functionality and less on ethical values related to the use of communication technologies [34]. Cross-domain IoT platforms can also advertise good QoS metrics, including privacy. Still, these metrics might not be credible, especially when profitability is threatened [231], and In [115], researchers stressed the need for IoT design and management frameworks that implement mechanisms for tangibly assessing privacy.

iv) Legal challenges: This case is particularly crucial when the composite system’s requirements include non-repudiation [36]. This goes against the users’ or devices’ privacy which is an "Element to Protect".

- **b) Privacy solutions in service composition**

Three solutions were pointed out to address privacy challenges in service composition:

i) Service and component-based solutions : By implementing components and services that enable, manage, and assess privacy within composition platforms. In [29], an IoT-A-based IoT composition platform (Fiware) leveraged the "Identity Management Functional Component (FC)" within the "Security Functional Group" to issue pseudonyms and manage accessory information to trusted subjects to ensure anonymous operations and as a result protect privacy. In [36], an Identity Management component run by a trusted third party was leveraged for user privacy protection and tracking malicious actions. Similarly, the CompoSecReasoner [115] component addressed privacy through the computation and validation of privacy metrics post-composition. Privacy was computed/estimated based on tangible vulnerability, exposure, and disclosure metrics.

ii) Best practices, Standards, and Regulatory solutions: this solution is illustrated in [38][39], where the NIST CPS Framework provides privacy risk management guidelines, including the analysis of privacy risks throughout the entire data lifecycle: creation, collection, composition, exploitation, and disposal. In [176] and [219], The EU's GPDR is proposed as a framework for addressing privacy by enforcing mechanisms of **Transparency** (how data are processed), **Consent** (user's ability to opt-in or opt-out), and **Erasure** (the right of the users to delete data). Human-Centric Computing, proposed in [34], also proposed architectural best practices for developing privacy-aware composition platforms from the development phase, and in [176][34], user consent was highlighted as an important component to protect users' privacy when accessing IoT cloud platforms. Similarly, researchers in [231] proposed a history record-based service optimization method (HireSomeII) that protects cross-clouds privacy by evaluating their QoS history records instead of relying on their advertised QoS values, which would enhance the credibility of the composition plans they provide. Another effort that tackled the problem of protecting user privacy in cloud platforms while enhancing other QoS parameters was mentioned in [254], where researchers used privacy-preserving mechanisms to rank compositions based on their privacy preservation level. Other best practices that enhance privacy in IoT composition platforms were discussed in [166], where researchers highlighted recommendations including the anonymization of personal data, encrypting and securing data storage components, and implementing user-customizable data sharing mechanisms

iii) Technology-based solutions: Including **blockchain, encryption, and cryptography** as discussed in [175] and [176], where IoT data was encrypted to preserve privacy in an ML/Blockchain-based smart-city composition platform, and in a cloud IoT platform. Blockchain has also been used in [175] and [176] to protect privacy by either storing sensitive data on a distributed ledger, making it difficult to trace or by leveraging the decentralized nature of the blockchain to avoid the case of a single entity managing devices and stakeholders credentials. **Decomposition** of privacy into atomic properties

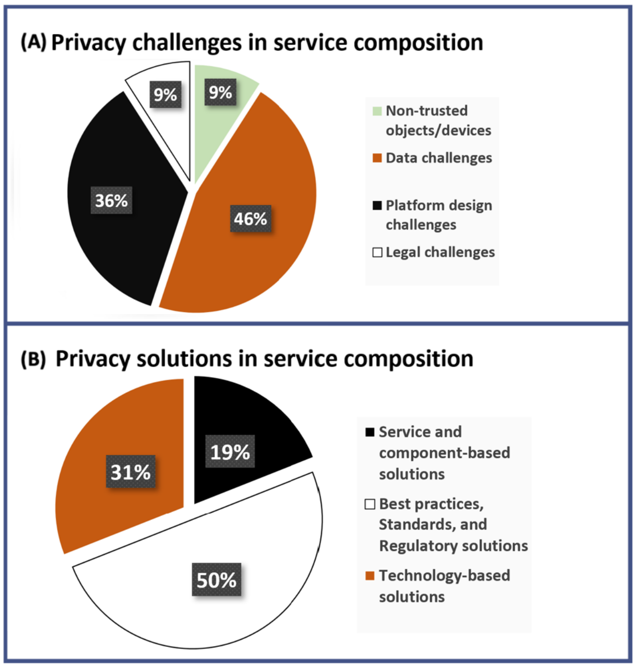

FIGURE 22. AQ15: privacy challenges and solutions in service composition

is another technical solution leveraged in [18] to address privacy concerns in IoT composition platforms. Researchers have decomposed privacy into atomic problems or properties, including authentication, authorization, data protection, unobservability, anonymity, unlinkability, undetectability, and pseudonymity. These low-level atomic properties were studied individually and thoroughly to improve privacy. Figure 22 shows trends related to IoT/CPS service composition privacy challenges (A) and solutions (B).

B. TRENDS, GAPS, AND THREATS TO VALIDITY

1) Trends

We can recognize what's trending in a certain topic and how important it is by running an advanced search -using different flavors of each sub-aspect's vocabulary- on the pool of primary studies (182 publications) using the Adobe Advanced Search tool.

The advanced search parses primary studies and searches for keywords related to the different taxonomy sub-aspects -mentions in the bibliography not considered- including some that were not addressed in this SLR. As opposed to the discussion section, we include in this analysis primary studies that not only discussed an aspect thoroughly but also papers that partially addressed it while discussing other sub-aspects. These trends are illustrated in Figure 23.

For the Formal aspect, almost 99% of primary studies mentioned a framework, a standard, or an architecture, which shows the importance of these components in guiding service composition. Although not addressed in this SLR, composition algorithms are a major aspect in the discussed primary studies, with more than 70% of primary studies mentioning at least one composition algorithm. Regarding the covered formal aspects in this SLR, we expect the problem of state space explosion to remain an important challenge in the upcoming

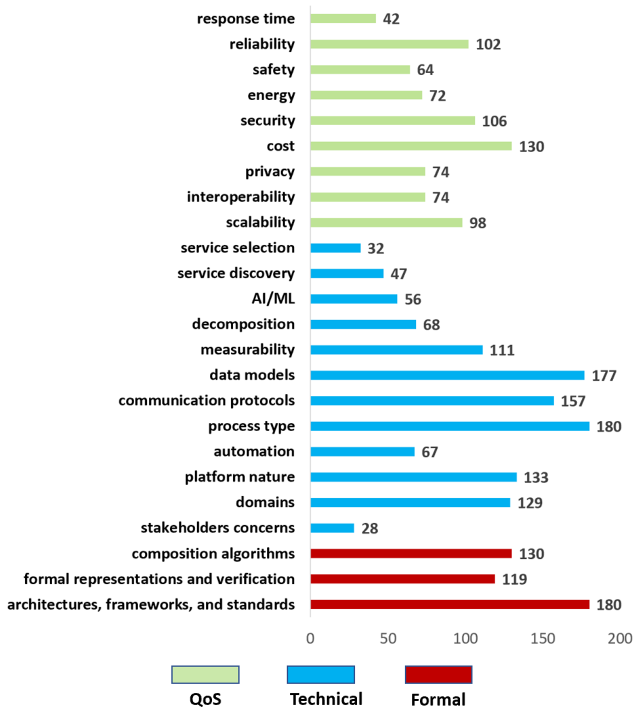

FIGURE 23. Primary studies trends in IoT/CPS capabilities composition and decomposition based on the number of primary studies that partially or thoroughly addressed a particular formal, technical, or QoS sub-aspect, including non-addressed sub-aspects in this study (Other Formal, Other Technical, Other QoS).

decade: in the absence of a substantial effort in optimizing composition algorithms, improvements in computation power will always lead to the innovation of capabilities with higher complexity, keeping the state space explosion problem a continuous concern for the research community.
For Technical sub-aspects, composition process type and data models are the most discussed aspects (more than 95% of primary studies), whereas service discovery and service selection were discussed in only 26% of primary studies or less. This is because we didn't pick primary studies based on these sub-aspects' keywords. For the upcoming years, content-centric networks such as NDN and CCN [161][162] are gaining momentum with their data caching and interest tracing capabilities, which could revolutionize service composition and decomposition in IoT platforms.
Finally, for the QoS aspect, cost, security, and reliability were mentioned in 56% of the primary studies; although they weren't the main subject of this study, this shows their importance and involvement when discussing other service composition sub-aspects.

2) Gaps

As mentioned in the trends section and based on the taxonomy proposed in this SLR study, many formal, technical, and QoS sub-aspects were not addressed through SLR questions but rather discussed partially within other sub-aspects.

For the Formal aspect, although composition algorithms were extensively discussed in [228] -specifically on how to leverage meta-heuristics algorithms to efficiently select capabilities based on user-defined QoS parameters-, service selection remains only one of many service composition steps. An SLR question that addresses how different service composition algorithms efficiently intervene during the other composition steps is a topic researchers need to invest effort in.

Regarding the Technical sub-aspects of service composition, automation and process types in service decomposition, and the level of automation of each step of the composition process are worth investigating by researchers, and we consider these topics as open issues.

For the QoS aspect, the security of composed systems was addressed in many non-SLR/Literature publications, including [115], where authors discussed the security of IoT systems of systems (SoS) and highlighted the importance of calculating the security level of the final/composed system, taking into consideration the security of its subsystems. However, security-specific SLR questions, such as the security mechanisms required during each composition or decomposition step, were not addressed in this SLR study or other SLR studies, making it a gap worth filling by the research community.

3) Threats to Validity

This paragraph presents possible threats to the SLR's validity while mentioning some corrective strategies.

For the document sources, only SCOPUS and Google Scholar databases were queried; however, SCOPUS alone generates results from more than 5000 publishers (including the main publishers), which should provide substantial results along with the search performed -for completion- in Google Scholar.
Regarding the SCOPUS and Google Scholar search strings: they were designed in a way that produces as many results as possible, with extra keywords added to filter specific sub-aspects questions. Not using all synonyms for a certain sub-aspect might result in missing a certain study. The search string automates the selection process as much as possible, but given the large number of papers and the different addressed questions, human error/bias during the non-automatic phases of the search process (manual selection, forward and backward snowballing) cannot be completely ruled out.

As for the selection procedure, it was partially automated as some stages require researchers' involvement and refinement (snowballing forward and backward and manual evaluation of the large number of initial results from both databases). The manual stages were repeated to minimize error and bias.

As for the possibility of false negatives, which could cause the exclusion of important studies, we ran the search strings multiple times and during multiple periods while conducting this study, which would reduce the chances of excluding important manuscripts.

As for the focus of the study, this SLR did not exclude non-academic efforts and cited not only scientific and academic publications but also industry solutions (especially in the platform type sub-aspect) to provide comprehensive results.

VI. CONCLUSION

In this section, we highlight this effort's main achievements, explain the study's benefits to different stakeholders, and share our thoughts on ongoing and future work.

A. MAIN ACHIEVEMENTS AND BENEFITS TO DIFFERENT STAKEHOLDERS.

As indicated in the abstract, we focused in this SLR on providing two contributions to the topic of IoT capabilities composition and decomposition: i) We contributed a reference taxonomy that researchers and future readers can use to identify/locate components of the topic relative to specific areas of research. The taxonomy frames the different facets of service composition in IoT in three aspects: A formal aspect, a technical aspect, and a QoS aspect. ii) We contributed detailed discussions to fill gaps in the knowledge corresponding to IoT capabilities composition and decomposition by addressing essential and unanswered research questions (RQs) related to the identified taxonomy aspects. Based on the provided answers, different stakeholders can leverage responses to RQs to improve the composition or decomposition processes, enabling better IoT platforms. For example, engineers, developers, and city planners will learn about the various aspects, challenges, and solutions related to the innovation of novel IoT capabilities composition and decomposition platforms. This effort provides valuable guidance on building novel IoT platforms while respecting formal and standard constraints, ensuring technical solutions used are the most suitable and efficient from network performance, process type, and automation perspective. Other benefits include highlighting the importance of AI/ML in improving service composition capabilities or processes and demonstrating the role of service decomposition in ensuring the efficient reuse of capabilities features or enabling efficient computation distribution from the cloud to edge nodes, to mention a few. QoS improvement is another venue where researchers and city planners can leverage this study to enhance scalability, interoperability, and privacy by learning from the challenges and solutions we identified for each QoS concern. This study educates end-users on how the composition of services drives innovation by generating value-added services and making them accessible to the general public, as is the case with well-known platforms, such as IFTTT and Home Assistant. In addition, it highlights general-public concerns that stem from exposing one's capabilities preferences, which could compromise end-users privacy.

B. THOUGHTS ON ONGOING AND FUTURE WORK

While working on this study, an IoT and CPS Composition Framework (ICCF) was proposed [193] to address the problem of rapid prototyping and verifying composite capabilities in the field of smart buildings, with the main contributions made on Formal sub-aspects.
Work on completing the proposed framework is ongoing by studying and improving its technical and QoS sub-aspects based on data in this SLR. Implementing the ICCF framework in the smart building domain was proposed, and measurability aspects were addressed as we suggested a method for measuring well-being or comfort in smart buildings [224].

NIST DISCLAIMER

Certain commercial equipment, instruments, or materials (or suppliers, software, etc.) are identified in this paper to foster understanding. Such identification does not imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the materials or equipment identified are necessarily the best available for the purpose. Official contribution of the National Institute of Standards and Technology; not subject to copyright in the United States.

REFERENCES

[1] C. Greer, M. Burns, D. Wollman, E. Griffor, et al., "Cyber-physical systems and internet of things," NIST Special Publication (SP) 1900-202, 2019.
[2] P. Asghari, A. M. Rahmani, and H. H. S. Javadi, "Service composition approaches in iot: A systematic review," Journal of Network and Computer Applications, vol. 120, pp. 61–77, 2018.
[3] A. Huf and F. Siqueira, "Composition of heterogeneous web services: A systematic review," Journal of Network and Computer Applications, vol. 143, pp. 89–110, 2019.
[4] I. Aoudia, S. Benharzallah, L. Kahloul, and O. Kazar, "Service composition approaches for internet of things: a review," International Journal of Communication Networks and Distributed Systems, vol. 23, no. 2, pp. 194–230, 2019.
[5] A. Souri, A. M. Rahmani, and N. Jafari Navimipour, "Formal verification approaches in the web service composition: a comprehensive analysis of the current challenges for future research," International journal of communication systems, vol. 31, no. 17, p. e3808, 2018.
[6] A. Vakili and N. J. Navimipour, "Comprehensive and systematic review of the service composition mechanisms in the cloud environments," Journal of Network and Computer Applications, vol. 81, pp. 24–36, 2017.
[7] K. Hofer-Schmitz and B. Stojanović, "Towards formal verification of iot protocols: A review," Computer Networks, vol. 174, p. 107233, 2020.
[8] V. M. Tayur and R. Suchithra, "Review of interoperability approaches in application layer of internet of things," in 2017 International Conference on Innovative Mechanisms for Industry Applications (ICIMIA), pp. 322–326, IEEE, 2017.
[9] M. S. Nikoo, Ö. Babur, and M. Van Den Brand, "A survey on service composition languages," in Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings, pp. 1–5, 2020.
[10] A. Souri and M. Norouzi, "A state-of-the-art survey on formal verification of the internet of things applications,"

Journal of Service Science Research, vol. 11, no. 1, pp. 47–67, 2019.

[11] C. Jatoth, G. Gangadharan, and R. Buyya, "Computational intelligence based qos-aware web service composition: a systematic literature review," IEEE Transactions on Services Computing, vol. 10, no. 3, pp. 475–492, 2015.

[12] B. Trammell, P. Casas, D. Rossi, A. Bär, Z. B. Houidi, I. Leontiadis, T. Szemethy, and M. Mellia, "mplane: an intelligent measurement plane for the internet," IEEE Communications Magazine, vol. 52, no. 5, pp. 148–156, 2014.

[13] O. Alsaryrah, I. Mashal, and T.-Y. Chung, "A fast iot service composition scheme for energy efficient qos services," in Proceedings of the 2019 7th International Conference on Computer and Communications Management, pp. 231–237, 2019.

[14] S. A. H. Bidi, Z. Movahedi, and Z. Movahedi, "Qoe-aware service composition in smart cities using restful iot," in Electrical Engineering (ICEE), Iranian Conference on, pp. 1559–1564, IEEE, 2018.

[15] S. Ahmad, D. Kim, et al., "Design and implementation of thermal comfort system based on tasks allocation mechanism in smart homes," Sustainability, vol. 11, no. 20, p. 5849, 2019.

[16] H. Yamaoka, K. Itakura, E. Takahashi, G. Nakagawa, J. Michaelis, Y. Kanemasa, M. Ueki, T. Matsumoto, R. Take, S. Tanie, et al., "Dracena: A real-time iot service platform based on flexible composition of data streams," in 2019 IEEE/SICE International Symposium on System Integration (SII), pp. 596–601, IEEE, 2019.

[17] M. Blackstock and R. Lea, "Iot mashups with the wotkit," in 2012 3rd IEEE International Conference on the Internet of Things, pp. 159–166, IEEE, 2012.

[18] M. Papoutsakis, K. Fysarakis, G. Spanoudakis, S. Ioannidis, and K. Koloutsou, "Towards a collection of security and privacy patterns," Applied Sciences, vol. 11, no. 4, p. 1396, 2021.

[19] B. Familiar, Microservices, IoT, and Azure. Springer, 2015.

[20] F. Cirillo, D. Straeten, D. Gomez, J. Gato, L. Diez, I. E. Maestro, and R. Akhavan, "Atomic services: sustainable ecosystem of smart city services through pan-european collaboration," in 2019 Global IoT Summit (GIoTS), pp. 1–7, IEEE, 2019.

[21] F. Durán, G. Salaün, and A. Krishna, "Automated composition, analysis and deployment of iot applications," in International Conference on Objects, Components, Models and Patterns, pp. 252–268, Springer, 2019.

[22] L. Zhang, S. Yu, X. Ding, and X. Wang, "Research on iot restful web service asynchronous composition based on bpel," in 2014 Sixth International Conference on Intelligent Human-Machine Systems and Cybernetics, vol. 1, pp. 62–65, IEEE, 2014.

[23] T. V. Atanasova, S. A. Poryazov, and E. T. Saranova, "Problems with quality enabling of information functions composition in smart buildings," in 2016 24th Telecommunications Forum (TELFOR), pp. 1–4, IEEE, 2016.

[24] I. Ullah, M. Sohail Khan, and D. Kim, "Iot services and virtual objects management in hyperconnected things network," Mobile Information Systems, vol. 2018, 2018.

[25] B. Trammell, M. Mellia, A. Finamore, S. Traverso, T. Szemethy, B. Szabo, D. Rossi, B. Donnet, F. Invernizzi, and D. Papadimitriou, "mplane architecture specification."

[26] A. Krishna, M. Le Pallec, R. Mateescu, L. Noirie, and G. Salaün, "Rigorous design and deployment of iot applications," in 2019 IEEE/ACM 7th International Conference on Formal Methods in Software Engineering (FormaliSE), pp. 11–20, IEEE, 2019.

[27] A. Krishna, M. Le Pallec, R. Mateescu, L. Noirie, and G. Salaün, "Iot composer: Composition and deployment of iot applications," in 2019 IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings (ICSE-Companion), pp. 19–22, IEEE, 2019.

[28] C. Newcombe, T. Rath, F. Zhang, B. Munteanu, M. Brooker, and M. Deardeuff, "How amazon web services uses formal methods," Communications of the ACM, vol. 58, no. 4, pp. 66–73, 2015.

[29] A. Preventis, K. Stravoskoufos, S. Sotiriadis, and E. G. Petrakis, "Iot-a and fiware: Bridging the barriers between the cloud and iot systems design and implementation.," in CLOSER (2), pp. 146–153, 2016.

[30] C. Mahmoudi, F. Mourlin, and A. Battou, "Formal definition of edge computing: An emphasis on mobile cloud and iot composition," in 2018 Third International Conference on Fog and Mobile Edge Computing (FMEC), pp. 34–42, IEEE, 2018.

[31] B. Alturki, S. Reiff-Marganiec, C. Perera, and S. De, "Exploring the effectiveness of service decomposition in fog computing architecture for the internet of things," IEEE Transactions on Sustainable Computing, 2019.

[32] F. Mohr, M. Wever, E. Hüllermeier, and A. Faez, "(wip) towards the automated composition of machine learning services," in 2018 IEEE International Conference on Services Computing (SCC), pp. 241–244, IEEE, 2018.

[33] F. Mohr, M. Wever, and E. Hüllermeier, "Automated machine learning service composition," arXiv preprint arXiv:1809.00486, 2018.

[34] D. Guinard, V. Trifa, F. Mattern, and E. Wilde, "From the internet of things to the web of things: Resource-oriented architecture and best practices," in Architecting the Internet of things, pp. 97–129, Springer, 2011.

[35] V. Nundloll, Y. Elkhatib, A. Elhabbash, and G. S. Blair, "An ontological framework for opportunistic composition of iot systems," in 2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT), pp. 614–621, IEEE, 2020.

[36] A. Bassi, M. Bauer, M. Fiedler, R. van Kranenburg, S. Lange, S. Meissner, and T. Kramp, Enabling things to talk. Springer Nature, 2013.

[37] B. Jazayeri and S. Schwichtenberg, "On-the-fly computing meets iot markets—towards a reference architecture," in 2017 IEEE International Conference on Software Architecture Workshops (ICSAW), pp. 120–127, IEEE, 2017.

[38] E. R. Griffor, C. Greer, D. A. Wollman, and M. J. Burns, "Framework for cyber-physical systems: Volume 1, overview," 2017.

[39] E. R. Griffor, C. Greer, D. A. Wollman, and M. J. Burns, "Framework for cyber-physical systems: Volume 2, working group reports," 2017.

[40] D. A. Wollman, M. A. Weiss, Y. Li-Baboud, E. R. Griffor, and M. J. Burns, "Framework for cyber-physical systems: Volume 3, timing annex," 2017.

[41] S. Yun, H. Kim, H. Shin, H. S. Chin, and W.-T. Kim, "A novel reference model for cloud manufacturing cps platform based on onem2m standard," KIPS Transactions on Computer and Communication Systems, vol. 8, no. 2, pp. 41–56, 2019.

[42] F. Sivrikaya, N. Ben-Sassi, X.-T. Dang, O. C. Görür, and C. Kuster, "Internet of smart city objects: A distributed framework for service discovery and composition," IEEE Access, vol. 7, pp. 14434–14454, 2019.

[43] G. N. Rai, G. Gangadharan, and V. Padmanabhan, "Algebraic modeling and verification of web service composition," Procedia computer science, vol. 52, pp. 675–679, 2015.

[44] G. N. Rai, G. Gangadharan, V. Padmanabhan, and R. Buyya, "Web service interaction modeling and verification using recursive composition algebra," IEEE Transactions on Services

Computing, 2018.
[45] D. Arellanes and K.-K. Lau, "Decentralized data flows in algebraic service compositions for the scalability of iot systems," in 2019 IEEE 5th World Forum on Internet of Things (WF-IoT), pp. 668–673, IEEE, 2019.
[46] D. Arellanes and K.-K. Lau, "Evaluating iot service composition mechanisms for the scalability of iot systems," Future Generation Computer Systems, 2020.
[47] D. Arellanes and K.-K. Lau, "Algebraic service composition for user-centric iot applications," in International Conference on Internet of Things, pp. 56–69, Springer, 2018.
[48] L. Li, Z. Jin, G. Li, L. Zheng, and Q. Wei, "Modeling and analyzing the reliability and cost of service composition in the iot: A probabilistic approach," in 2012 IEEE 19th International Conference on Web Services, pp. 584–591, IEEE, 2012.
[49] H. Zhao, W. Wang, J. Sun, and Y. Wei, "Research on formal modeling and verification of bpel-based web service composition," in 2012 IEEE/ACIS 11th International Conference on Computer and Information Science, pp. 631–636, IEEE, 2012.
[50] Mernik, M. Formal and Practical Aspects of Domain-Specific Languages: Recent Developments: Recent Developments. (IGI global,2012)
[51] R. Xiao, Z. Wu, and D. Wang, "A finite-state-machine model driven service composition architecture for internet of things rapid prototyping," Future Generation Computer Systems, vol. 99, pp. 473–488, 2019.
[52] Y. Li, S. Zhao, H. Diao, and H. Chen, "A formal validation method for trustworthy services composition," in 2016 International Conference on Networking and Network Applications (NaNA), pp. 433–437, IEEE, 2016.
[53] A. Krishna, M. L. Pallec, A. Martinez, R. Mateescu, and G. Salaün, "Mozart: Design and deployment of advanced iot applications," in Companion Proceedings of the Web Conference 2020, pp. 163–166, 2020.
[54] H. Mkaouar, B. Zalila, J. Hugues, and M. Jmaiel, "From aadl model to lnt specification," in Ada-Europe International Conference on Reliable Software Technologies, pp. 146–161, Springer, 2015.
[55] D. Champelovier, X. Clerc, H. Garavel, Y. Guerte, F. Lang, C. McKinty, V. Powazny, W. Serwe, and G. Smeding, "Reference manual of the lnt to lotos translator," 2018.
[56] W. L. Yeung, "A formal and visual modeling approach to choreography based web services composition and conformance verification," Expert Systems with Applications, vol. 38, no. 10, pp. 12772–12785, 2011.
[57] M. Zhu, J. Li, G. Fan, and K. Zhao, "Modeling and verification of response time of qos-aware web service composition by timed csp," Procedia Computer Science, vol. 141, pp. 48–55, 2018.
[58] B. Li, S. Ji, D. Qiu, H. Leung, and G. Zhang, "Verifying the concurrent properties in bpel based web service composition process," IEEE Transactions on Network and Service Management, vol. 10, no. 4, pp. 410–424, 2013.
[59] C. Laneve and L. Padovani, "An algebraic theory for web service contracts," Formal Aspects of Computing, vol. 27, no. 4, pp. 613–640, 2015.
[60] A. Souri, A. M. Rahmani, N. J. Navimipour, and R. Rezaei, "A hybrid formal verification approach for qos-aware multi-cloud service composition," Cluster Computing, vol. 23, no. 4, pp. 2453–2470, 2020.
[61] T. Baker, M. Asim, H. Tawfik, B. Aldawsari, and R. Buyya, "An energy-aware service composition algorithm for multiple cloud-based iot applications," Journal of Network and Computer Applications, vol. 89, pp. 96–108, 2017.
[62] N. Le Sommer, Y. Mahéo, and F. Baklouti, "Multi-strategy dynamic service composition in opportunistic networks," Information, vol. 11, no. 4, p. 180, 2020.
[63] H. Groefsema, N. R. van Beest, and M. Aiello, "A formal model for compliance verification of service compositions," IEEE Transactions on Services Computing, vol. 11, no. 3, pp. 466–479, 2016.
[64] C. Simpkin, I. Taylor, D. Harborne, G. Bent, A. Preece, and R. K. Ganti, "Dynamic distributed orchestration of node-red iot workflows using a vector symbolic architecture," in 2018 IEEE/ACM Workflows in Support of Large-Scale Science (WORKS), pp. 52–63, IEEE, 2018.
[65] A. Souri and M. Ghobaei-Arani, "Cloud manufacturing service composition in iot applications: a formal verification-based approach," Multimedia Tools and Applications, pp. 1–20, 2021.
[66] A. Krishna, 2019, "Compose and build iot applications using lnt models." https://github.com/ajaykrishna/iotcomposer.(Accessed 2022)
[67] A. Krishna, 2020, "Advanced iot compositions in webthings gateway." https://github.com/ajaykrishna/mozart, (Accessed 2022).
[68] P. Valderas, V. Torres, and V. Pelechano, "A microservice composition approach based on the choreography of bpmn fragments," Information and Software Technology, vol. 127, p. 106370, 2020.
[69] P. Valderas, 2020, "Microservice composition based on the choreography of bpmn fragments." https://github.com/pvalderas/microservices-composition-example, (Accessed 2022).
[70] D. Hershcovich, 2015, "modelling composition for directed acyclic graphs in python." https://github.com/ninjin/nerv, (Accessed 2022).
[71] D. Arellanes, 2019, "Dx-man platform for building iot systems using algebraic service composition." https://github.com/damianarellanes/dxman, (Accessed 2022).
[72] S. Sedelnikov, 2020, ".net implementation of the finite state machine framework." https://github.com/serge-sedelnikov/xstate.net, (Accessed 2022).
[73] F. Mohr, 2020, "Mlplan." https://github.com/fmohr/ML-Plan, (Accessed 2022).
[74] K. Bae, P. C. Ölveczky, T. H. Feng, E. A. Lee, and S. Tripakis, "Verifying hierarchical ptolemy ii discrete-event models using real-time maude," Science of Computer Programming, vol. 77, no. 12, pp. 1235–1271, 2012.
[75] A. Iannopollo, 2017, "Pyco ltl composition." https://github.com/ianno/pyco, (Accessed 2022).
[76] J. Harlow, 2020, "A library for communicating sequential processes in node.js, built on top of async/await." https://github.com/f5io/csp, (Accessed 2022).
[77] M. Halter, 2020 "Build petri net models compositionally." https://github.com/AlgebraicJulia/AlgebraicPetri.jl, (Accessed 2022).
[78] E. Fares, J.-P. Bodeveix, and M. Filali, "Event algebra for transition systems composition-application to timed automata," in 2013 20th International Symposium on Temporal Representation and Reasoning, pp. 125–132, IEEE, 2013.
[79] A. Bacciu, 2019 "An sml library for calculus of communicating systems." https://github.com/andreabac3/Calculus-of-Communicating-Systems, (Accessed 2022).
[80] M. Kwiatkowska, G. Norman, and D. Parker, "Prism 4.0: Verification of probabilistic real-time systems," in International conference on computer aided verification, pp. 585–591, Springer, 2011.
[81] I. Sarray, A. Ressouche, D. Gaffé, J.-Y. Tigli, and S. Lavirotte, "Safe composition in middleware for the internet of

things," in Proceedings of the 2nd Workshop on Middleware for Context-Aware Applications in the IoT, pp. 7–12, 2015.
[82] A. Gatouillat and Y. Badr, "Verifiable and resource-aware component model for iot devices," in Proceedings of the 9th International Conference on Management of Digital EcoSystems, pp. 235–242, 2017.
[83] Inria, 2021 "Coq." //https://github.com/coq/coq, (Accessed 2022).
[84] F. Kammüller, "Human centric security and privacy for the iot using formal techniques," in International Conference on Applied Human Factors and Ergonomics, pp. 106–116, Springer, 2017.
[85] K. Kappelmann, 2022, "Isabelle theorem prover" https://github.com/isabelle-prover, (Accessed 2022).
[86] P. Poizat and G. Salaün, "Checking the realizability of bpmn 2.0 choreographies," in Proceedings of the 27th Annual ACM Symposium on Applied Computing, pp. 1927–1934, 2012.
[87] G. Salaün, "Quantifying the similarity of non-bisimilar labelled transition systems," Science of Computer Programming, vol. 202, p. 102580, 2021.
[88] L. Lamport, 2021 "Tla+." https://github.com/tlaplus/tlaplus, (Accessed 2022).
[89] "Cadp online request form." https://cadp.inria.fr/registration. (Accessed 2022).
[90] "Isabelle installation guide." https://isabelle.in.tum.de/installation.html. (Accessed 2022).
[91] "The coq assistant installation guide." https://coq.inria.fr/download. (Accessed 2022).
[92] S. Ouchani, K. Khebbeb, and M. Hafsi, "Towards enhancing security and resilience in cps: A coq-maude based approach," in 2020 IEEE/ACS 17th International Conference on Computer Systems and Applications (AICCSA), pp. 1–6, IEEE, 2020.
[93] "Mwb'99 tool installation." https://www.it.uu.se/research/group /concurrency/mwb/installation. (Accessed 2022).
[94] A. Belghiat and A. Chaoui, "A pi-calculus-based approach for the verification of uml2 sequence diagrams," in 2015 10th International Joint Conference on Software Technologies (ICSOFT), vol. 2, pp. 1–8, IEEE, 2015.
[95] "Cwb-nc download folder." http://sens.cse.msu.edu/Software/CWB-NC/. (Accessed 2022).
[96] M. El-Menshawy, J. Bentahar, W. El Kholy, and R. Dssouli, "Verifying conformance of multi-agent commitment-based protocols," Expert Systems with Applications, vol. 40, no. 1, pp. 122–138, 2013.
[97] "Spin download page." http://spinroot.com/spin/Src/index.html. (Accessed 2022).
[98] S. Chouali, A. Boukerche, A. Mostefaoui, and M. A. Merzoug, "Formal verification and performance analysis of a data exchange protocol for connected vehicles," IEEE Transactions on Vehicular Technology, 2020.
[99] "Maude download and installation." http://maude.cs.illinois.edu/w/index.php. (Accessed 2022).
[100] H. Merouani, F. Mokhati, and H. Seridi-Bouchelaghem, "Towards formalizing web service composition in maude's strategy language," in Proceedings of the 1st International Conference on Intelligent Semantic Web-Services and Applications, pp. 1–6, 2010.
[101] "Uppaal download and installation." https://uppaal.org/downloads/. (Accessed 2022).
[102] Y. Lu and M. Sun, "Modeling and verification of ieee 802.11 i security protocol in uppaal for internet of things," International Journal of Software Engineering and Knowledge Engineering, vol. 28, no. 11n12, pp. 1619–1636, 2018.
[103] "Nusmv download and installation guide." https://nusmv.fbk.eu/NuSMV/download/getting-v2.html. (Accessed 2022).
[104] S. Agarwal and M. Dubey, "Checking of web services composition through online movie ticket booking example," 2017.
[105] A. Kheldoun and M. Ioualalen, "Transformation bpel processes to recatnet for analysing web services compositions," in 2014 2nd International Conference on Model-Driven Engineering and Software Development (MODELSWARD), pp. 425–430, IEEE, 2014.
[106] M. Al-Diabat, F. Al-Saqqar, A. Al-Saggar, B. Uinverity, and J. Mafraq, "Automatic verification of communicative commitments using reduction,"
[107] N. Adadi, M. Berrada, D. Chenouni, and B. Bounabat, "Proposition of web services composition approach basing of model-driven approach and multi-agent systems," International Journal of Computer Modelling & New Technologies, 2017.
[108] F. Chevrou, A. Hurault, and P. Quéinnec, "Automated verification of asynchronous communicating systems with tla+," Electronic Communications of the EASST, vol. 72, pp. pp–1, 2015.
[109] M. A. Kuppe, L. Lamport, and D. Ricketts, "The tla+ toolbox," arXiv preprint arXiv:1912.10633, 2019.
[110] H. Wang, Q. Zhou, and Y. Shi, "Describing and verifying web service composition using tla reasoning," in 2010 IEEE International Conference on Services Computing, pp. 234–241, IEEE, 2010.
[111] H. Garavel, F. Lang, R. Mateescu, and W. Serwe, "Cadp 2011: a toolbox for the construction and analysis of distributed processes," International Journal on Software Tools for Technology Transfer, vol. 15, no. 2, pp. 89–107, 2013.
[112] V. Molnár, B. Graics, A. Vörös, I. Majzik, and D. Varró, "The gamma statechart composition framework: Design, verification and code generation for component-based reactive systems," in 2018 IEEE/ACM 40th International Conference on Software Engineering: Companion (ICSE-Companion), pp. 113–116, IEEE, 2018.
[113] S. Agarwal and K. Agarwal, "Pi-calculus based formal verification of web services composition," International Journal of Grid and Distributed Computing, vol. 8, no. 5, pp. 137–140, 2015.
[114] Y. Du, W. Tan, and M. Zhou, "Timed compatibility analysis of web service composition: A modular approach based on petri nets," IEEE Transactions on Automation Science and Engineering, vol. 11, no. 2, pp. 594–606, 2013.
[115] G. Hatzivasilis, N. Papadakis, I. Hatzakis, S. Ioannidis, and G. Vardakis, "Artificial intelligence-driven composition and security validation of an internet of things ecosystem," Applied Sciences, vol. 10, no. 14, p. 4862, 2020.
[116] R. Maroui and B. Ayeb, "Integrating the sysml and acme in a model driven engineering approach to verify the web service composition," in 2015 IEEE 24th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises, pp. 183–189, IEEE, 2015.
[117] C. Akasiadis, G. Tzortzis, E. Spyrou, and C. Spyropoulos, "Developing complex services in an iot ecosystem," in 2015 IEEE 2nd World Forum on Internet of Things (WF-IoT), pp. 52–56, IEEE, 2015.
[118] K. Halba, E. Griffor, P. Kamongi, and T. Roth, "Using statistical methods and co-simulation to evaluate ads-equipped vehicle trustworthiness," in 2019 Electric Vehicles International Conference (EV), pp. 1–5, IEEE, 2019.
[119] D. Lu, D. Huang, A. Walenstein, and D. Medhi, "A secure microservice framework for iot," in 2017 IEEE Symposium on Service-Oriented System Engineering (SOSE), pp. 9–18,

IEEE, 2017.
[120] F. Cirillo, D. Gómez, L. Diez, I. E. Maestro, T. B. J. Gilbert, and R. Akhavan, "Smart city iot services creation through large scale collaboration," IEEE Internet of Things Journal, 2020.
[121] N. Kalatzis, N. Marianos, and F. Chatzipapadopoulos, "Iot and data interoperability in agriculture: A case study on the gaiasense tm smart farming solution," in 2019 Global IoT Summit (GIoTS), pp. 1–6, IEEE, 2019.
[122] B. Cheng, G. Solmaz, F. Cirillo, E. Kovacs, K. Terasawa, and A. Kitazawa, "Fogflow: Easy programming of iot services over cloud and edges for smart cities," IEEE Internet of Things Journal, vol. 5, no. 2, pp. 696–707, 2017.
[123] A. Urbieta, A. González-Beltrán, S. B. Mokhtar, M. A. Hossain, and L. Capra, "Adaptive and context-aware service composition for iot-based smart cities," Future Generation Computer Systems, vol. 76, pp. 262–274, 2017.
[124] N. Xie, W. Tan, X. Zheng, L. Zhao, L. Huang, and Y. Sun, "An efficient two-phase approach for reliable collaboration-aware service composition in cloud manufacturing," Journal of Industrial Information Integration, vol. 23, p. 100211, 2021.
[125] Z. Liu, L. Wang, X. Li, and S. Pang, "A multi-attribute personalized recommendation method for manufacturing service composition with combining collaborative filtering and genetic algorithm," Journal of Manufacturing Systems, vol. 58, pp. 348–364, 2021.
[126] J. Spillner, "Quantitative analysis of cloud function evolution in the aws serverless application repository," arXiv preprint arXiv:1905.04800, 2019.
[127] Amazon Web Services, 2021, "Green grass iot." https://github.com/aws-greengrass, (Accessed 2022).
[128] Zhou, J., Athukorala, K., Gilman, E., Riekki, J. & Yliant-tila, M. Cloud architecture for dynamic service composition. International Journal Of Grid And High Performance Computing (IJGHPC). **4**, 17-31 (2012)
[129] F. Lemoine, T. Aubonnet, and N. Simoni, "Iot composition based on self-controlled services," Journal of Ambient Intelligence and Humanized Computing, vol. 11, no. 11, pp. 5167–5186, 2020.
[130] "Create an IoT hub using the Azure portal, howpublished = https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-create-through-portal, note = Accessed 2022."
[131] Microsoft, 2021 "Azure iot." https://github.com/Azure/iot, (Accessed 2022).
[132] A. Bhattacharjee, Y. Barve, A. Gokhale, and T. Kuroda, "(wip) cloudcamp: Automating the deployment and management of cloud services," in 2018 IEEE International Conference on Services Computing (SCC), pp. 237–240, IEEE, 2018.
[133] A. Bhattacharjee, 2021 "Automated designing and provisioning platform for cloud applications." https://github.com/doc-vu/DeploymentAutomation, (Accessed 2022).
[134] fp7mplane, 2015, "mplane python3 sdk." https://github.com/fp7mplane/protocol-ri, (Accessed 2022).
[135] F. Invernizzi, 2015, "mplane node js implementation." https://github.com/finvernizzi/mplane, (Accessed 2022).
[136] Roth, T., Lemieux, C. & Burns, M. The UCEF Approach to Tool Integration for HLA Co-Simulations. 2020 IEEE Workshop On Design Automation For CPS And IoT (DESTION). pp. 10-18 (2020)
[137] K. Halba, 2021 "Ucef ads testbed." https://github.com/usnistgov/avchallenge, (Accessed 2022).
[138] S. Kokh, 2020, A. Fiol, J. Booth, and D. Dunajski, "Node-RED and ibm watson integration." https://github.com/acoustic-content-samples/sample-node-red-contrib-wch, (Accessed 2022).
[139] F. Jacob, E. Kevin, T. Alexander, and M. Rohit, 2021, "Ifttt github repository." https://github.com/IFTTT, (Accessed 2022).
[140] Y. Wang, L. Huang, X. Liu, T. Sun, and K. Lei, "Performance comparison and evaluation of websocket frameworks: Netty, undertow, vert. x, grizzly and jetty," in 2018 1st IEEE International Conference on Hot Information-Centric Networking (HotICN), pp. 13–17, IEEE, 2018.
[141] K. Halba, 2021,"IoT Capabilities Composition Framework" https://github.com/usnistgov/ICCF, (Accessed 2022).
[142] F. Nijhof, P. Sinclair, and D. Higgs, 2021,"Home assistant add-ons." https://github.com/hassio-addons/hassio-vagrant, (Accessed 2022).
[143] A. Krishna, 2019, "Iot composer." https://www.github.com/ajaykrishna/iotcomposer.git, (Accessed 2022).
[144] "Installing Fiware IoT Broker, howpublished = https://fiware-iotagent-ul.readthedocs.io/en/1.10.0/installationguide/, note = Accessed 2022."
[145] "A PLUGIN FOR COMPOSING ENTITIES FROM OTHER ENTITIES, howpublished = https://fiware-iot-broker.readthedocs.io/en/latest/entitycomposition/, note = Accessed 2022."
[146] A. Kazmi, M. Serrano, and J. Soldatos, "Vital-os: An open source iot operating system for smart cities," IEEE Communications Standards Magazine, vol. 2, no. 2, pp. 71–77, 2018.
[147] F. Heimgaertner, S. Hettich, O. Kohlbacher, and M. Menth, "Scaling home automation to public buildings: A distributed multiuser setup for openhab 2," in 2017 Global Internet of Things Summit (GIoTS), pp. 1–6, IEEE, 2017.
[148] B. Ur, M. Pak Yong Ho, S. Brawner, J. Lee, S. Mennicken, N. Picard, D. Schulze, and M. L. Littman, "Trigger-action programming in the wild: An analysis of 200,000 ifttt recipes," in Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems, pp. 3227–3231, 2016.
[149] M. Noura and M. Gaedke, "Wotdl: web of things description language for automatic composition," in 2019 IEEE/WIC/ACM International Conference on Web Intelligence (WI), pp. 413–417, IEEE, 2019.
[150] Rodrigues, C., Afonso, J. & Tomé, P. Mobile application webservice performance analysis: Restful services with json and xml. International Conference On ENTERprise Information Systems. pp. 162-169 (2011)
[151] I. Al Ridhawi, M. Aloqaily, A. Boukerche, and Y. Jaraweh, "A blockchain-based decentralized composition solution for iot services," in Icc 2020-2020 ieee international conference on communications (icc), pp. 1–6, IEEE, 2020.
[152] W. Li, G. Privat, J. M. Cantera, M. Bauer, and F. Le Gall, "Graph-based semantic evolution for context information management platforms," in 2018 Global Internet of Things Summit (GIoTS), pp. 1–6, IEEE, 2018.
[153] C.-A. Sun, Y. Zhao, L. Pan, H. Liu, and T. Y. Chen, "Automated testing of ws-bpel service compositions: A scenario-oriented approach," IEEE Transactions on Services Computing, vol. 11, no. 4, pp. 616–629, 2015.
[154] A. J. Maâlej and M. Krichen, "Study on the limitations of ws-bpel compositions under load conditions," The Computer Journal, vol. 58, no. 3, pp. 385–402, 2015.
[155] P. Marwaha, H. Banati, and P. Bedi, "Formalizing bpel-tc through [pi]-calculus," International Journal of Web & Semantic Technology, vol. 4, no. 3, p. 11, 2013.
[156] A. J. Maâlej, M. Lahami, M. Krichen, and M. Jmaïel, "Distributed and resource-aware load testing of ws-bpel compositions.," in ICEIS (2), pp. 29–38, 2018.
[157] W. Wang, S. De, R. Toenjes, E. Reetz, and K. Moessner, "A comprehensive ontology for knowledge representation in the internet of things," in 2012 IEEE 11th International

Conference on Trust, Security and Privacy in Computing and Communications, pp. 1793–1798, IEEE, 2012.
[158] P. Barnaghi, F. Ganz, H. Abangar, M. Presser, and K. Moessner, "Sense2web: A linked data platform for semantic sensor networks," Semantic Web-Interoperability, Usability, Applicability an IOS Press Journa, vol. 2, no. 1, pp. 1–11, 2011.
[159] Y.-Y. Fanjiang, Y. Syu, C.-H. Wu, J.-Y. Kuo, and S.-P. Ma, "Genetic algorithm for qos-aware dynamic web services composition," in 2010 International Conference on Machine Learning and Cybernetics, vol. 6, pp. 3246–3251, IEEE, 2010.
[160] W. Jin, R. Xu, S. Lim, D.-H. Park, C. Park, and D. Kim, "Integrated service composition approach based on transparent access to heterogeneous iot networks using multiple service providers," Mobile Information Systems, vol. 2021, 2021.
[161] Q. Wang, B. Lee, N. Murray, and Y. Qiao, "Cs-man: Computation service management for iot in-network processing," in 2016 27th Irish Signals and Systems Conference (ISSC), pp. 1–6, IEEE, 2016.
[162] S. Khoussi, A. Nouri, J. Shi, J. Filliben, L. Benmohamed, A. Battou, and S. Bensalem, "Performance evaluation of the ndn data plane using statistical model checking," in International symposium on automated technology for verification and analysis, pp. 534–550, Springer, 2019.
[163] M. Farnaghi and A. Mansourian, "Automatic composition of wsmo based geospatial semantic web services using artificial intelligence planning," Journal of Spatial Science, vol. 58, no. 2, pp. 235–250, 2013.
[164] M. Hosseinzadeh, Q. T. Tho, S. Ali, A. M. Rahmani, A. Souri, M. Norouzi, and B. Huynh, "A hybrid service selection and composition model for cloud-edge computing in the internet of things," IEEE Access, vol. 8, pp. 85939–85949, 2020.
[165] M. Sun, Z. Zhou, J. Wang, C. Du, and W. Gaaloul, "Energy-efficient iot service composition for concurrent timed applications," Future Generation Computer Systems, vol. 100, pp. 1017–1030, 2019.
[166] J. Soldatos, Building Blocks for IoT Analytics. River Publishers, 2016.
[167] H. Wang, P. Ma, and X. Zhou, "A quantitative and qualitative approach for nfp-aware web service composition," in 2012 IEEE Ninth International Conference on Services Computing, pp. 202–209, IEEE, 2012.
[168] B. Donassolo, I. Fajjari, A. Legrand, and P. Mertikopoulos, "Fog based framework for iot service provisioning," in 2019 16th IEEE Annual Consumer Communications & Networking Conference (CCNC), pp. 1–6, IEEE, 2019.
[169] M. M. Gomes, R. d. R. Righi, and C. A. da Costa, "Future directions for providing better iot infrastructure," in Proceedings of the 2014 ACM international joint conference on pervasive and ubiquitous computing: Adjunct Publication, pp. 51–54, 2014.
[170] W. Lv, F. Meng, C. Zhang, Y. Lv, N. Cao, and J. Jiang, "A general architecture of iot system," in 2017 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC), vol. 1, pp. 659–664, IEEE, 2017.
[171] A. Balalaie, A. Heydarnoori, P. Jamshidi, D. A. Tamburri, and T. Lynn, "Microservices migration patterns," Software: Practice and Experience, vol. 48, no. 11, pp. 2019–2042, 2018.
[172] K. Baek and I.-Y. Ko, "Spatio-cohesive service selection using machine learning in dynamic iot environments," in International Conference on Web Engineering, pp. 366–374, Springer, 2018.
[173] P. Grace, J. Barbosa, B. Pickering, and M. Surridge, "Taming the interoperability challenges of complex iot systems," in Proceedings of the 1st ACM Workshop on Middleware for Context-Aware Applications in the IoT, pp. 1–6, 2014.
[174] V. Chang and M. Ramachandran, "Towards achieving data security with the cloud computing adoption framework," IEEE Transactions on services computing, vol. 9, no. 1, pp. 138–151, 2015.
[175] M. Shen, X. Tang, L. Zhu, X. Du, and M. Guizani, "Privacy-preserving support vector machine training over blockchain-based encrypted iot data in smart cities," IEEE Internet of Things Journal, vol. 6, no. 5, pp. 7702–7712, 2019.
[176] G. Lee and U. Jayasinghe, "Ai and blockchain enabled edge of things with privacy preserving computation," in Conference Proceedings of The XIII International scientific conference Perspective technologies in the information transfer means, pp. 39–43, 2019.
[177] A. Safaei, R. Nassiri, and A. M. Rahmani, "Enterprise service composition models in iot context: solutions comparison," The Journal of Supercomputing, pp. 1–28, 2021.
[178] G. J. Ibrahim, T. A. Rashid, and M. O. Akinsolu, "An energy efficient service composition mechanism using a hybrid meta-heuristic algorithm in a mobile cloud environment," Journal of Parallel and Distributed Computing, vol. 143, pp. 77–87, 2020.
[179] Z.-y. Chai, M.-m. Du, and G.-z. Song, "A fast energy-centered and qos-aware service composition approach for internet of things," Applied Soft Computing, vol. 100, p. 106914, 2021.
[180] K. Alwasel, D. N. Jha, F. Habeeb, U. Demirbaga, O. Rana, T. Baker, S. Dustdar, M. Villari, P. James, E. Solaiman, et al., "Iotsim-osmosis: A framework for modeling and simulating iot applications over an edge-cloud continuum," Journal of Systems Architecture, vol. 116, p. 101956, 2021.
[181] O. Almurshed, O. Rana, Y. Li, R. Ranjan, D. N. Jha, P. Patel, P. P. Jayaraman, and S. Dustdar, "A fault tolerant workflow composition and deployment automation iot framework in a multi cloud edge environment," IEEE Internet Computing, pp. 1–1, 2021.
[182] B. Zhang, K. Wen, J. Lu, and M. Zhong, "A top-k qos-optimal service composition approach based on service dependency graph," Journal of Organizational and End User Computing (JOEUC), vol. 33, no. 3, pp. 50–68, 2021.
[183] R. Esmaeilyfard and M. Naderi, "Distributed composition of complex event services in iot network," The Journal of Supercomputing, vol. 77, no. 6, pp. 6123–6144, 2021.
[184] S. S. Sefati and N. J. Navimipour, "A qos-aware service composition mechanism in the internet of things using a hidden markov model-based optimization algorithm," IEEE Internet of Things Journal, 2021.
[185] F. Chen, C. Ren, J. Dong, Q. Wang, J. Li, and B. Shao, "Modeling cross-organizational services composition with pi-calculus," in Proceedings of 2011 IEEE International Conference on Service Operations, Logistics and Informatics, pp. 51–56, IEEE, 2011.
[186] D. Arellanes, 2019, "Parse a simple custom description language for labeled transition systems, create composites (synchronize on identical transition symbols), and graph them.." https://github.com/marvk/Labelled-Transition-System, (Accessed 2022).
[187] E. M. Clarke, W. Klieber, M. Nováček, and P. Zuliani, "Model checking and the state explosion problem," in LASER Summer School on Software Engineering, pp. 1–30, Springer, 2011.
[188] B. Blanchet, B. Smyth, V. Cheval, and M. Sylvestre, "Proverif 2.00: automatic cryptographic protocol verifier, user manual and tutorial," Version from, pp. 05–16, 2018.
[189] H. Wang, J. Li, Q. Yu, T. Hong, J. Yan, and W. Zhao, "Inte-

grating recurrent neural networks and reinforcement learning for dynamic service composition," Future Generation Computer Systems, vol. 107, pp. 551–563, 2020.
[190] F. Dai, Q. Mo, Z. Qiang, B. Huang, W. Kou, and H. Yang, "A choreography analysis approach for microservice composition in cyber-physical-social systems," IEEE Access, vol. 8, pp. 53215–53222, 2020.
[191] Z. Maamar, M. Asim, S. Cheikhrouhou, and A. Qamar, "Orchestration-and choreography-based composition of internet of transactional things," Service Oriented Computing and Applications, vol. 15, no. 2, pp. 157–170, 2021.
[192] J. Friesen, Java XML and JSON. Springer, 2016.
[193] Halba, K., Griffor, E., Lbath, A. & Dahbura, A. A Framework for the Composition of IoT and CPS Capabilities. 2021 IEEE 45th Annual Computers, Software, And Applications Conference (COMPSAC). pp. 1265-1272 (2021)
[194] C. Mahmoudi, 2017, " UPPAAL model for Mobile Cloud (Cloudlets) verification " https://github.com/charifmahmoudi/CloudletModel, (Accessed 2022).
[195] Khani Dehnoi, M. & Araban, S. A systematic literature review on web services composition. International Journal Of Nonlinear Analysis And Applications. **13**, 2821-2855 (2022)
[196] Hayyolalam, V., Pourghebleh, B., Chehrehzad, M. & Pourhaji Kazem, A. Single-objective service composition methods in cloud manufacturing systems: Recent techniques, classification, and future trends. Concurrency And Computation: Practice And Experience. **34**, e6698 (2022)
[197] Lei, Y., MingYuan, Y. & Zhang, H. A Domain-oriented Business-Standardization Service Organization Model. 2010 International Conference On Computer Application And System Modeling (ICCASM 2010). **1** pp. V1-666 (2010)
[198] Liu, J. & Tong, W. Adaptive service framework based on grey decision-making in the internet of things. 2010 6th International Conference On Wireless Communications Networking And Mobile Computing (WiCOM). pp. 1-4 (2010)
[199] Gutiérrez-Garcıéa, J., Ramos-Corchado, F. & Koning, J. Obligation-based agent conversations for semantic web service composition. 2009 IEEE/WIC/ACM International Joint Conference On Web Intelligence And Intelligent Agent Technology. **1** pp. 411-417 (2009)
[200] Liu, J. & Tong, W. Dynamic services model based on context resources in the internet of things. 2010 6th International Conference On Wireless Communications Networking And Mobile Computing (WiCOM). pp. 1-4 (2010)
[201] Papazoglou, M., Traverso, P., Dustdar, S. & Leymann, F. Service-oriented computing: a research roadmap. International Journal Of Cooperative Information Systems. **17**, 223-255 (2008)
[202] Song, Z., Cárdenas, A. & Masuoka, R. Semantic middleware for the internet of things. 2010 Internet Of Things (IOT). pp. 1-8 (2010)
[203] Souza, L., Spiess, P., Guinard, D., Köhler, M., Karnouskos, S. & Savio, D. Socrades: A web service based shop floor integration infrastructure. The Internet Of Things. pp. 50-67 (2008)
[204] Alam, S., Chowdhury, M. & Noll, J. Senaas: An event-driven sensor virtualization approach for internet of things cloud. 2010 IEEE International Conference On Networked Embedded Systems For Enterprise Applications. pp. 1-6 (2010)
[205] Da, Z., Bo, C., Yang, Z. & Junliang, C. Future service provision: Towards a flexible hybrid service supporting platform. 2010 IEEE Asia-Pacific Services Computing Conference. pp. 226-233 (2010)
[206] Zhou, H., Huang, Z. & Zhao, G. A service-centric solution for wireless sensor networks. 2010 5th International ICST Conference On Communications And Networking In China. pp. 1-5 (2010)
[207] Santhanam, G., Basu, S. & Honavar, V. On utilizing qualitative preferences in web service composition: a cp-net based approach. 2008 IEEE Congress On Services-Part I. pp. 538-544 (2008)
[208] Atzori, L., Iera, A. & Morabito, G. The internet of things: A survey. Computer Networks. **54**, 2787-2805 (2010)
[209] Karnouskos, S., Guinard, D., Savio, D., Spiess, P., Baecker, O., Trifa, V. & Souza, L. Towards the real-time enterprise: service-based integration of heterogeneous SOA-ready industrial devices with enterprise applications. IFAC Proceedings Volumes. **42**, 2131-2136 (2009)
[210] Moritz, G., Cornelius, C., Golatowski, F., Timmermann, D. & Stoll, R. Differences and commonalities of service-oriented device architectures, wireless sensor networks and networks-on-chip. 2009 International Conference On Advanced Information Networking And Applications Workshops. pp. 482-487 (2009)
[211] Guinard, D., Trifa, V., Pham, T. & Liechti, O. Towards physical mashups in the web of things. 2009 Sixth International Conference On Networked Sensing Systems (INSS). pp. 1-4 (2009)
[212] Hwang, S., Wang, H., Tang, J. & Srivastava, J. A probabilistic approach to modeling and estimating the QoS of web-services-based workflows. Information Sciences. **177**, 5484-5503 (2007)
[213] Sheng, Q., Qiao, X., Vasilakos, A., Szabo, C., Bourne, S. & Xu, X. Web services composition: A decade's overview. Information Sciences. **280** pp. 218-238 (2014)
[214] Lee, K., Fu, M. & Kuo, Y. A hierarchical scheduling strategy for the composition services architecture based on cloud computing. The 2nd International Conference On Next Generation Information Technology. pp. 163-169 (2011)
[215] Ghobaei-Arani, M. & Souri, A. LP-WSC: a linear programming approach for web service composition in geographically distributed cloud environments. The Journal Of Supercomputing. **75**, 2603-2628 (2019)
[216] Pathak, S. & Pandey, M. Smart cities: Review of characteristics, composition, challenges and technologies. 2021 6th International Conference On Inventive Computation Technologies (ICICT). pp. 871-876 (2021)
[217] Lemos, A., Daniel, F. & Benatallah, B. Web service composition: a survey of techniques and tools. ACM Computing Surveys (CSUR). **48**, 1-41 (2015)
[218] Friedewald, M., Vildjiounaite, E., Punie, Y. & Wright, D. The brave new world of ambient intelligence: An analysis of scenarios regarding privacy, identity and security issues. International Conference On Security In Pervasive Computing. pp. 119-133 (2006)
[219] Tragos, E., Bernabe, J., Staudemeyer, R., Luis, J., Ramos, H., Fragkiadakis, A., Skarmeta, A., Nati, M. & Gluhak, A. Trusted IoT in the complex landscape of governance, security, privacy, availability and safety. Digitising The Industry-Internet Of Things Connecting The Physical, Digital And Virtual Worlds. River Publishers Series In Communications. pp. 210-239 (2016)
[220] Gherabi, N. & Bahaj, M. A new method for mapping UML class into OWL ontology. Spec. Issue Int. J. Comput. Appl.(0975 8887) Soft. Eng. Databases Expert Syst. SEDEXS. pp. 5-9 (2012)
[221] Jazayeri, B. Architectural management of on-the-fly computing markets. Proccedings Of The 10th European Conference On Software Architecture Workshops. pp. 1-2 (2016)
[222] Karl, H., Kundisch, D., Heide, F. & Wehrheim, H. A case for a new IT ecosystem: On-The-Fly computing. Business & Information Systems Engineering. **62**, 467-481 (2020)
[223] Huma, Z., Gerth, C. & Engels, G. On-The-Fly computing:

automatic service discovery and composition in heterogeneous domains. Computer Science-Research And Development. **30**, 333-361 (2015)

[224] Halba, K., Hailane, A., Lbath, A., Griffor, E. & Dahbura, A. Well-being as a Composite Capability in the Smart Building Domain: A Formal and Technical Study. 2022 IEEE European Technology And Engineering Management Summit (E-TEMS). pp. 115-125 (2022)

[225] Happe, M., Heide, F., Kling, P., Platzner, M. & Plessl, C. On-the-fly computing: A novel paradigm for individualized it services. 16th IEEE International Symposium On Object/component/service-oriented Real-time Distributed Computing (ISORC 2013). pp. 1-10 (2013)

[226] Kitchenham, B. & Charters, S. Guidelines for performing systematic literature reviews in software engineering. (Citeseer,2007)

[227] Garcıéa Holgado, A., Marcos Pablos, S., Garcıéa Peñalvo, F. & Others Guidelines for performing systematic research projects reviews. International Journal Of Interactive Multimedia And Artificial Intelligence. **6**, 9 (2020)

[228] Naghavipour, H., Soon, T., Idris, M., Namvar, M., Salleh, R. & Gani, A. Hybrid Metaheuristics for QoS-aware Service Composition: A Systematic Mapping Study. IEEE Access. (2021)

[229] Kitchenham, B., Pretorius, R., Budgen, D., Brereton, O., Turner, M., Niazi, M. & Linkman, S. Systematic literature reviews in software engineering–a tertiary study. Information And Software Technology. **52**, 792-805 (2010)

[230] Keele, S. & Others Guidelines for performing systematic literature reviews in software engineering. (Technical report, Ver. 2.3 EBSE Technical Report. EBSE,2007)

[231] Dou, W., Zhang, X., Liu, J. & Chen, J. HireSome-II: Towards privacy-aware cross-cloud service composition for big data applications. IEEE Transactions On Parallel And Distributed Systems. **26**, 455-466 (2013)

[232] Lin, L., Hu, J. & Zhang, J. Packet: a privacy-aware access control policy composition method for services composition in cloud environments. Frontiers Of Computer Science. **10**, 1142-1157 (2016)

[233] Badii, C., Bellini, P., Difino, A. & Nesi, P. Smart city IoT platform respecting GDPR privacy and security aspects. IEEE Access. **8** pp. 23601-23623 (2020)

[234] Brereton, P., Kitchenham, B., Budgen, D., Turner, M. & Khalil, M. Lessons from applying the systematic literature review process within the software engineering domain. Journal Of Systems And Software. **80**, 571-583 (2007)

[235] Zschaler, S. Formal specification of non-functional properties of component-based software systems. Software and Systems Modeling. **9**, 161-201 (2010)

[236] Celik, Z., McDaniel, P. & Tan, G. Soteria: Automated IoT Safety and Security Analysis. 2018 USENIX Annual Technical Conference (USENIX ATC 18). pp. 147-158 (2018)

[237] Arellanes, D. & Lau, K. D-XMAN: a platform for total compositionality in service-oriented architectures. 2017 IEEE 7th International Symposium On Cloud And Service Computing (SC2). pp. 283-286 (2017)

[238] Sun, Y., McMillin, B., Liu, X. & Cape, D. Verifying noninterference in a cyber-physical system the advanced electric power grid. Seventh International Conference On Quality Software (QSIC 2007). pp. 363-369 (2007)

[239] Nimble Code, 2022, " An Efficient Logic Model Checker for the Verification of Multi-threaded Code " https://github.com/nimble-code/Spin, (Accessed 2022).

[240] Stephan. Merz, Markus. Alexander Kuppe, 2021,"PlusCal Cheat Sheet by Stephan Merz" https://github.com/tlaplus/PlusCalCheatSheet,(Accessed 2022).

[241] Dal Lago, L., Ferrante, O., Passerone, R. & Ferrari, A. Dependability assessment of SOA-based CPS with contracts and model-based fault injection. IEEE Transactions On Industrial Informatics. **14**, 360-369 (2017)

[242] Niu, D., Li, Y., Zhang, Z. & Song, B. A Service Composition Mechanism Based on Mobile Edge Computing for IoT. 2019 6th International Conference On Information Science And Control Engineering (ICISCE). pp. 982-985 (2019)

[243] D. Kerkkamp, 2020, " Learning State Representations for Formal Verification " https://github.com/davidkerkkamp/representation-learning, (Accessed 2022).

[244] Paraskevopoulou, Z., Li, J. & Appel, A. Compositional optimizations for certicoq. Proceedings Of The ACM On Programming Languages. **5**, 1-30 (2021)

[245] Spichkova, M. & Schmidt, H. Towards logical architecture and formal analysis of dependencies between services. 2014 Asia-Pacific Services Computing Conference. pp. 121-128 (2014)

[246] Z. Paraskevopoulou, 2022, et al., " CertiCoq " https://github.com/CertiCoq/certicoq, (Accessed 2022).

[247] Wenzel, M. Miscellaneous Isabelle/Isar examples. (Accessed 2022)

[248] Safri, H., Kandi, M., Miloudi, Y., Bortolaso, C., Trystram, D. & Desprez, F. A Federated Learning Framework for IoT: Application to Industry 4.0. 2022 22nd IEEE International Symposium On Cluster, Cloud And Internet Computing (CCGrid). pp. 565-574 (2022)

[249] Rajendran, R., Sharma, P., Saran, N., Ray, S., Alanya-Beltran, J. & Tongkachok, K. An Exploratory analysis of Machine Learning adaptability in Big Data Analytics Environments: A Data Aggregation in the age of Big Data and the Internet of Things. 2022 2nd International Conference On Innovative Practices In Technology And Management (ICIPTM). **2** pp. 32-36 (2022)

[250] Mendonça, F., Dantas, M., Fortunato, W., Oliveira, J., Souza, B. & Filgueiras, M. Wearable Devices in Healthcare: Challenges, Current Trends and a Proposition of Affordable Low Cost and Scalable Computational Environment of Internet of Things. Brazilian Congress On Biomedical Engineering. pp. 1301-1308 (2022)

[251] Nisansala, S., Chandrasiri, G., Prasadika, S. & Jayasinghe, U. Microservice Based Edge Computing Architecture for Internet of Things. 2022 2nd International Conference On Advanced Research In Computing (ICARC). pp. 332-337 (2022)

[252] Tatemura, J. & Hsiung, W. Web service decomposition: Edge computing architecture for cache-friendly e-commerce applications. Electronic Commerce Research And Applications. **5**, 57-65 (2006)

[253] Gorige, D., Al-Masri, E., Kanzhelev, S. & Fattah, H. Privacy-risk detection in microservices composition using distributed tracing. 2020 IEEE Eurasia Conference On IOT, Communication And Engineering (ECICE). pp. 250-253 (2020)

[254] Asghari, P., Rahmani, A. & Javadi, H. Privacy-aware cloud service composition based on QoS optimization in Internet of Things. Journal Of Ambient Intelligence And Humanized Computing. pp. 1-26 (2020)

[255] Kaldeli, E., Warriach, E., Bresser, J., Lazovik, A. & Aiello, M. Interoperation, composition and simulation of services at home. International Conference On Service-Oriented Computing. pp. 167-181 (2010)

[256] Antonić, A., Marjanović, M. & Žarko, I. Modeling aggregate input load of interoperable smart city services. Proceedings Of The 11th ACM International Conference On Distributed And Event-Based Systems. pp. 34-43 (2017)

[257] K. Halba, et al., 2022, " SLR Primary Studies Data " https://github.com/KhalidHALBA/SLR, (Accessed 2022).

KHALID HALBA (M'19) received a Master's of Engineering in Networks and Telecommunications from Mohammadia Engineering School (EMI) in Rabat, Morocco, in 2014. From 2014 to 2015, he worked as a Telecommunication Engineer at AFD Technologies in Rabat, Morocco, where he analyzed KPIs and generated recommendations on Ericsson and Huawei LTE stations located in France. He joined the National Institute of Standards and Technology (NIST) from 2015 to 2020 in Gaithersburg, Maryland, USA, as a Guest Researcher. He worked in NIST's ITL and EL labs, where he published contributions in the fields of Software-Defined In-Vehicle Networks, Co-simulation techniques, and IoT/CPS capabilities composition. Mr. HALBA is pursuing a Ph.D. in computer science at the Grenoble Alpes University (UGA) in Grenoble, France. In 2018, he acquired the Cisco CCNA Certification. In 2020, He was awarded the NIST/HOPKINS PREP fellowship and joined Johns Hopkins University (JHU) in Baltimore, Maryland, USA, as an Assistant Research Scientist, and was promoted in 2022 to an Associate Research Scientist at JHU, where he is currently working in the Institute of Assured Autonomy (IAA).

EDWARD GRIFFOR is the Associate Director for Cyber-Physical Systems in the Engineering Laboratory at the U.S. National Institute of Standards and Technology. He is Chair of the MIT Alliance and leads US Government efforts on IoT and system safety and security and Formal Methods for System Assurance. He received his Ph.D. in Mathematics from MIT and holds a "Habilitation in Electrical Engineering". Dr. Griffor has research collaborations with government and university laboratories in the UK, Sweden, and France. His research interests include cyber-physical systems and Internet of Things interoperability. He holds multiple patents and has published numerous books, journals, and conference proceedings, including the Handbook of Computability, the Handbook of System Safety and Security, and the Mathematical Theory of Domains. Contact him at edward.griffor@nist.gov.

AHMED LBATH is a Distinguished Full Professor of computer science at Grenoble Alpes University (UGA) in Grenoble, France. He is the Director of the Carnot Institute of Intelligent Systems and Software. He received his Ph.D. degree in computer science from the University of INSA Lyon and holds a "Habilitation à Diriger des Recherches". He is member of the Board of Thrustees of the University of Grenoble, and President of the Finance Committee. He is Associate on cyber-physical systems at the French Embassy of the United States at the Department of Science and Technology. He is also conducting research in collaboration with EL/ITL Labs at NIST in the Washington DC metro area where he carried out research activities as a visiting professor. His research interests include cyber-physical human systems, smart cities, knowledge engineering, recommendation systems, GIS, and software design. He published several patents, papers in books, journals, and conferences, and he regularly serves as a co-chair and/or member of several committees of international conferences, journals, and research programs. Contact him at ahmed.lbath@imag.fr.

ANTON DAHBURA (M'81-SM'87-F'96) received the BSEE, MSEE, and PhD in Electrical Engineering and Computer Science from the Johns Hopkins University in 1981, 1982, and 1984, respectively. From 1983 to 1996, he was a researcher at AT&T Bell Laboratories, was an Invited Lecturer in the Department of Computer Science at Princeton University, and served as the Research Director of the Motorola Cambridge Research Center in Cambridge, Massachusetts. Since 1996, he has led several entrepreneurial efforts in the areas of commercial printing, professional baseball operations as co-owner of the Hagerstown Suns (Washington Nationals Class A affiliate) and commercial real estate. In January, 2012 he was named Executive Director of the Johns Hopkins University Information Security Institute in Baltimore and joined the faculty of the Johns Hopkins University Department of Computer Science as an Associate Research Scientist. He also holds an appointment at the Johns Hopkins University Malone Center for Engineering in Healthcare, and in September 2018, he was named Co-Director of the Johns Hopkins University Institute for Assured Autonomy.