A Systematic Mapping Study of Deployment and Orchestration Approaches for IoT

Phu H. Nguyen[1], Nicolas Ferry[1], Gencer Erdogan[1], Hui Song[1], Stéphane Lavirotte[2], Jean-Yves Tigli[2] and Arnor Solberg[3]
[1]*SINTEF Digital, Oslo, Norway*
[2]*Université Nice Sophia Antipolis, Nice, France*
[3]*Tellu IoT AS, Asker, Norway*
{*firstname.lastname*}*@sintef.no* {*firstname.lastname*}*@unice.fr* {*firstname.lastname*}*@tellu.no*

Keywords: IoT, Review, Survey, Deployment, Orchestration, Trustworthiness

Abstract: IoT systems are typically distributed and perform coordinated behavior across IoT, edge and cloud infrastructures. It is very challenging to deploy and orchestrate IoT systems across different layers. We need a clear picture of the research landscape of the existing deployment and orchestration approaches for IoT (DEPO4IOT). Such a picture can show us how advanced the current state of the art is and what are the gaps to address. We conducted a systematic mapping study (SMS) to find out the research landscape in this area. The results of our SMS show the overall status of the key artifacts of DEPO4IOT. Among the results, we found a sharp increase in the number of primary DEPO4IOT publications in two recent years. We also found that most approaches do not really support the deployment or orchestration at low-level IoT devices. There is a lack of addressing the trustworthy aspects and advanced supports in the existing DEPO4IOT approaches. Finally, we point out the current open issues in this research area and suggest potential research directions to tackle these issues.

1 INTRODUCTION

The world is now on the verge of "the IoT age" with the Internet of Things (IoT) installed units predicted to grow to 20.4 billion by 2020[1]. IoT systems can be classified as systems of systems in which physical systems (a.k.a, "things") and cyber systems are combined and connected via connection means. Despite having enormous potential, the heterogeneous nature of IoT brings up great challenges that must be addressed to fully realize the potential of the IoT. In particular, because IoT systems are typically distributed and performing coordinated behavior across IoT, edge and cloud infrastructures (A. Metzger (Ed.), 2015), it is important to facilitate their creation and operation. In this context, research community and industry have been proposing different approaches and tools for supporting the deployment and/or orchestration of IoT systems. However, it is not clear what are the primary existing approaches for supporting the deployment and/or orchestration of IoT systems, and how advanced they are.

To provide a clear picture of the research landscape of the existing approaches and tools for supporting the deployment and/or orchestration of IoT systems (DEPO4IOT), we conducted a systematic mapping study (SMS). Our SMS has three main objectives. First, we want to summarize the existing primary DEPO4IOT approaches. Second, by analyzing the existing approaches, we can identify any gaps in the state-of-the-art. Moreover, we want to examine whether the DEPO4IOT approaches consider trustworthiness aspects such as security, privacy, resilience, reliability, and safety. Addressing trustworthiness aspects is particularly important because IoT systems can, for instance, have an impact on the physical world through actuators or can deal with sensitive data. The ability of a system to evolve, quickly react to failures and release patches is decisive to ensure and increase the trustworthiness of IoT systems. Therefore, we also want to find out if any approach has provided advanced supports such as monitoring and adaptation. Third, based on the results, we propose new research activities to fill the gaps for supporting modern IoT systems. We followed the latest guidelines in (Petersen et al., 2015) to conduct our SMS. We have systematically filtered thousands of relevant papers from four main on-line publication

[1]Gartner, February 7, 2017

databases, and a manual search process, to finally obtain a set of *sixty nine* (69) primary DEPO4IOT studies. We extracted and synthesized data from the primary studies to answer our *research questions* (RQs). The main contributions of this work are our answers to the following RQs. **RQ1**: *What are the publication statistics of the primary* DEPO4IOT *studies?* **RQ2**: *What are the primary* DEPO4IOT *approaches and how advanced are they, especially in addressing trustworthiness aspects?* **RQ3**: *What are the open issues to be further investigated in this field?*

In the remainder of this paper: Section 2 gives some background definitions. In Section 3, we present our SMS approach. To facilitate the data extraction and comparison, Section 4 describes our classification schemes for the primary studies. We present the results of our SMS in Section 5. Related work is discussed in Section 6. Finally, we conclude the paper with the major findings and some directions for future work in Section 7.

2 BACKGROUND

In this section, we give the definitions of IoT systems (subsection 2.1), orchestration and deployment (subsection 2.2), and trustworthiness (subsection 2.3) that were used to define the scope of this study.

2.1 Definition of IoT systems

An IoT system is "*a system of entities (including cyber-physical devices, information resources, and people) that exchange information and interact with the physical world by sensing, processing information, and actuating*" (IEEE Standards Association et al., 2016). In this work, we consider DEPO4IOT approaches that are independent or specific to an IoT application domain such as smart city or health-care.

2.2 Definition of Deployment and Orchestration

Software deployment typically refers to a post-development activity performed once a piece of software has been developed to make it available for use (Carzaniga et al., 1998). In this study, approaches are considered as supporting deployment when they explicitly offer mechanisms enabling the software deployment process, which typically consists of the following stages: (i) release, (ii) installation, (iii) activation, (iv) update, (v) adaptation, and (vi) undeployment (Dearie, 2007).

Deployment approaches typically rely on the concept of software artifacts or components as a modular part of a system that encapsulates its contents (Dearie, 2007). A deployment configuration is thus a connected graph that describes deployment artifacts along with targets and relationships between them from a structural perspective (Bergmayr et al., 2018). In the cloud context, the term "topology" is often used as well. A deployment configuration or topology typically includes the description of how its software components are integrated and communicate with each other. This typically overlaps with the concept of software orchestration (or composition) because an *orchestration* is also often represented as a graph describing relationships between software elements or processes. However, contrary to deployment configurations, these relationships may represent an order of operations required to realize a behavior. Using this definition, any service composition approach that targets IoT domain is considered as an orchestration approach for IoT.

2.3 Definition of Trustworthiness

Trustworthiness refers to the preservation of security, privacy, safety, reliability, and resilience of systems (Griffor et al., 2017). *Security* refers to the preservation of confidentiality, integrity and availability of information (IEC, ISO, 2012). *Privacy* refers to the protection of personally identifiable information (PII) (IEC, ISO, 2011). *Safety* refers to the ability of the systems to ensure the absence of catastrophic consequences on the life, health, property, or data of stakeholders and the physical environment (Griffor et al., 2017). *Reliability* refers to the ability of the systems to deliver stable and predictable performance in expected conditions (Griffor et al., 2017). *Resilience* refers to the ability of the systems to withstand instability, unexpected conditions, and gracefully return to predictable, but possibly degraded, performance (Griffor et al., 2017).

3 OUR SYSTEMATIC MAPPING APPROACH

We conducted our SMS by following the latest guidelines for conducting SMS in (Petersen et al., 2015) and other relevant guidelines in (Kitchenham, 2004). With the context and motivation presented in Section 1, we define our RQs for this work in Section 3.1. To explicitly define the scope of our SMS and reduce possible bias in our selection process, in Section

3.2, we clarify the inclusion criteria for selecting primary studies. Section 3.3 shows our search strategy to find the primary studies for answering the RQs.

3.1 Research questions

We detail the general RQs given Section 1 into the sub-RQs as follows. RQ1 includes three sub-RQs. **RQ1.1 -** *In which years were the primary studies published and how many publications per year?* Answering this question allows us to know for how long this field has been started, and how often are there publications. This is an indicator to assess if this field has got more attention from the research community. **RQ1.2 -** *In which targeted venues (e.g., Software Engineering-SE, IoT, Cloud/SOA, and Network) and venue types (i.e., conference, journal, workshop) were the primary studies published?* Answering RQ1.2 would enable us to know which venues have been the targets for publication of primary studies, knowing that DEPO4IOT approaches could crosscut several related research areas such as SE, IoT, SOA, Cloud. The venue types could also provide some hints on the maturity of the primary studies, *i.e.*, papers published at journals are supposed to report more mature studies than papers published at conferences and workshops. **RQ1.3 -** *What is the distribution of publications in terms of academic and industrial affiliation?* A paper is classified as *academia* if all authors come from university or research institute, and *industry* if all authors come from a company, and *both* (academia and industry) if there is a mix of authors from academia and industry. Answering RQ1.3 would give us an indication of how much collaboration between academia and industry on this topic as well as on the interest and need for DEPO4IOT approaches in the industry.

RQ2 is broken down into the following sub-RQs. **RQ2.1:** *How do the primary* DEPO4IOT *approaches support IoT orchestration?* Answering RQ2.1 allows us to assess the characteristics of the primary studies in IoT orchestration techniques. **RQ2.2:** *How do the primary* DEPO4IOT *approaches support IoT deployment?* Answering RQ2.2 allows us to assess the characteristics of the primary studies in IoT deployment techniques. When answering RQ2.1 and RQ2.2, we also examine how the primary DEPO4IOT approaches support for deployment, orchestration of IoT systems across the vastly heterogeneous IoT, edge and cloud space. Particularly, we examine how trustworthiness aspects are supported by the primary DEPO4IOT studies. Trustworthiness aspects are important in the development and operation of IoT systems and are thus often checked in surveys of IoT (*e.g.*, (Tran et al., 2017)). **RQ2.3:** *Are there any primary approaches explicitly addressing trustworthiness aspects?* Answering RQ2.3 allows us to examine the support of the primary DEPO4IOT approaches towards trustworthy IoT systems. **RQ2.4:** *Are there (runtime) adaptation or shared access to resources mechanisms supported in the primary* DEPO4IOT *approaches?* Answering RQ2.4 allows us to check if any approaches have provided advanced supports such as adaptation or shared access to resources to increase IoT systems trustworthiness (see Section 4.3). **RQ2.5:** *How were the primary* DEPO4IOT *approaches evaluated?* Answering RQ2.5 allows us to know what evaluation methods have been used in the primary studies, *e.g.*, industrial case studies or empirical studies. How a research approach has been evaluated is an indicator of the maturity of the work, especially if its evaluation used industrial case studies or empirical studies.

Based on the answers to the RQ1 and RQ2, we can point out the open issues that would deserve more investigation in the future and some potential directions to tackle these issues. **RQ3** has two sub-RQs: **RQ3.1** - *What are the open issues of* DEPO4IOT *research?* **RQ3.2** - *What research directions could be recommended for tackling the open issues?*

3.2 Inclusion and exclusion criteria

Based on the RQs and the scope of our study presented in Section 1, we clearly predefined the inclusion and exclusion criteria to reduce bias in our process of search and selection of primary studies. The primary studies must meet ALL the following inclusion criteria (IC):
- (IC1) Must propose a deployment OR orchestration approach.
- (IC2) Must be explicitly for IoT area, either in general or in a specific application domain of IoT.
- (IC3) Must have software engineering approaches as software is the main drive of deployment and orchestration for IoT.

We excluded non-peer-reviewed or unpublished paper, white paper, technical report, thesis, patent, general web page, presentation, book chapter, and paper not written in English.

3.3 Search and selection strategy

Fig. 1 shows an overview of the search and selection process with the results for each step, which we describe as follows.

3.3.1 Database search

Using the online search functions of popular publication databases is the most common way to search for

primary studies when conducting secondary studies (Petersen et al., 2015). We used four popular publication databases IEEE Xplore[2], ACM DL[3], Science Direct[4], and Scopus[5] to search for candidates of primary studies. We did not use Google Scholar and SpringerLink. Scopus and ACM DL already index SpringerLink[6] (Tran et al., 2017). Google Scholar returns all kinds of papers, in which peer-reviewed articles should have been covered by our four chosen databases. Worse, Google Scholar also returns many non-peer-reviewed and non-English papers, which should have been excluded at the first place. The four chosen databases contain peer-reviewed articles, and provide advanced search functions, especially search in meta-data such as title, abstract, keywords that we used. Based on the RQs, we identified the search keywords. Basically, we used the following search query: *("Internet of Things" OR IoT OR "Web of things" OR WoT) AND (orchestration OR deployment OR choreography OR topology OR composition OR dataflow) AND (Tool OR Middleware OR Service OR Framework)*. For each database, the search string needs to be applied according to the search functions provided.

For each candidate paper, we first read the paper's title, keywords and abstract. If a candidate paper appears in more than one database, we only kept the original one where it was published first. If the title, keywords and abstract are insufficient for us to decide to exclude a candidate paper, we further skimmed and scanned the paper's full content. We rather kept any candidate paper in doubt at one point for further checks later. In the end, we hold discussions among reviewers to crosscheck the candidate papers in doubt and agreed on including or excluding each paper. After group discussions, we obtained 63 primary studies.

3.3.2 Manual search

It is impractical to make sure the database search results can cover all DEPO4IOT publications in our study. Therefore, we tried to complement the database search by a manual search. We started our manual search by initiating a set of the DEPO4IOT studies that we have known of such as the studies numbered 22, 24, 26, 30, 31, 35, 42, 49, 50, 53, 55 in Table 1. This was also the test set to fine-tune the search query in the database search. Moreover, we checked the latest work of the authors of these DEPO4IOT studies and their related work to find more DEPO4IOT studies (using Google scholar). We found six new primary studies that have not been found in the database search (because the common keywords are not in their titles or abstracts). In total, we obtained the final set of 69 primary studies[7] for data extraction and synthesis to answer the RQs.

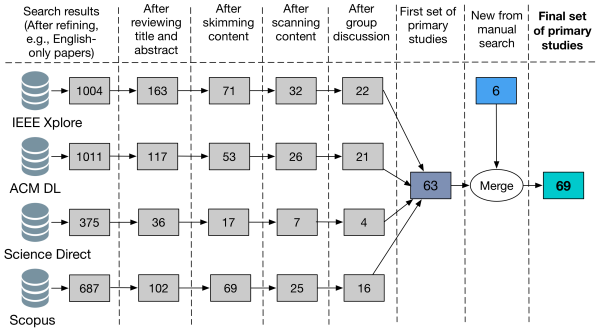

Figure 1: Overview of the search and selection steps

4 A TAXONOMY FOR CLASSIFICATION

To analyze the primary studies for answering our RQs, we have defined a taxonomy as a classification and comparison framework for the primary studies. Our taxonomy consists of the key aspects of deployment and orchestration for IoT (Fig. 2). We group them into the following categories: deployment and orchestration support (Section 4.1), design support (Section 4.2), and advanced support (Section 4.3).

4.1 Deployment and orchestration support

4.1.1 Deployment support

Different deployment approaches can be classified by deployment kind, target, network specification, whether cloud provisioning is supported, and what kind of bootstrap is used in deployment process.

Deployment kind: Automatic deployment typically comes in two flavors: imperative and declarative (Binz et al., 2013). The imperative approach requires a "deployment plan" that details how to reach the desired deployment, usually using a workflow language. This deployment plan defines a sequence of atomic

[2] http://ieeexplore.ieee.org
[3] https://dl.acm.org
[4] https://www.sciencedirect.com
[5] https://www.scopus.com
[6] https://www.springer.com/gp/computer-science/lncs/information-on-abstracting-and-indexing/799288

[7] Our search and selection process for the primary studies ended on 16 March 2018

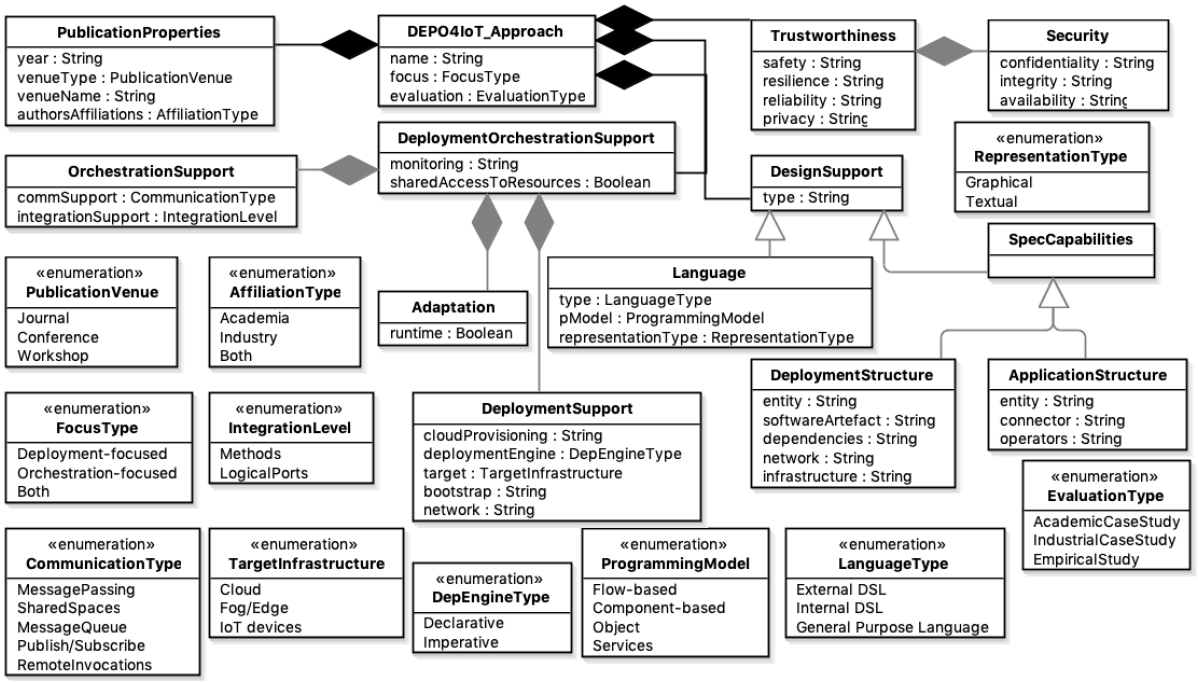

Figure 2: A Taxonomy of Deployment and Orchestration for IoT

tasks to execute possibly in parallel. This imperative approach gives full control over the deployment process, but it remains difficult to specify and reuse such plans. In contrast, the declarative approach only requires a specification of the desired deployment. This state usually captures the needed application structure and deployment configuration using a domain-specific languages (DSL) and a "deployment engine" then computes how to reach this state. This approach better supports evolving and reusing topology models, however the deployment engine may not compute optimal plans.

Target: IoT systems are typically running over heterogeneous infrastructures ranging from tiny devices over gateways to cloud resources. Following the layered architecture presented in (Bonomi et al., 2012) we divide this infrastructure into three layers, starting from the most constrained: (i) the device layer, which typically consists of embedded devices, sensors and actuators with low power and bandwidth; (ii) the edge/fog layer, which consists of the infrastructure located at the edge of the network who bridge the gap between the tiny devices and the cloud; and (iii) the cloud layer, which offers a virtually infinite set of computing and storage resources. The latter is commonly categorized based on the layers of virtualization (Armbrust et al., 2010): infrastructure (IaaS), platform (PaaS), and software (SaaS).

Network: Orchestration and deployment approaches must consider networking aspects to ensure that connected software components, deployed over the whole IoT, edge and cloud space, can properly interact with each other (*e.g.*, custom addressing and segmentation of launched Virtual Machines - VMs).

Cloud provisioning: One key characteristic of a modern cloud environment is the capability to dynamically provision cloud resources (Mell and Grance, 2001). Cloud users can provision and release cloud services on demand and pay only for what they have actually consumed. This includes compute, storage and network resources. It is not only important to distinguish tools that actually support cloud resources provisioning from others but also to identify which virtualization layer (i.e., IaaS, PaaS, SaaS) their provisioning engine supports.

Bootstrap: This characterizes the kind of bootstrap the solution relies on. A bootstrap is a basic executable program on a device, or a runtime environment, which the system in charge of the deployment relies on (*e.g.*, Docker). This aspect relates to the specificity of the solution and its dependency to particular technologies (Arcangeli et al., 2015).

4.1.2 Orchestration support

Orchestration approaches can be classified by kinds of communication and integration.

Communication support: we classify five different kinds of communication support in orches-

tration (Eugster et al., 2003): MessagePassing, SharedSpaces, MessageQueue, Publish-Subscribe, and RemoteInvocations (*e.g.*, Java RMI, CORBA). *MessagePassing*: the producer sends messages asynchronously through a communication channel. The consumer receives messages by listening synchronously on that channel. *Shared space*: Producers insert data asynchronously into the shared space, while consumers read data synchronously. *Message queuing*: Messages are stored in a FIFO queue. Producers append messages asynchronously at the end of the queue, while consumers dequeue them synchronously at the front of the queue. *Publish-Subscribe*: subscribers have the ability to express their interest in an event, or a pattern of events, and are subsequently notified of any event, generated by a publisher, which matches their registered interest.

Integration support: The interactions between the software components that compose an orchestration or a deployment can be specified and managed at different levels of abstraction. Deployment tools typically manage logical ports (*e.g.*, port 22 should be open for SSH) for setting up the communication channel between the software components to be deployed. Orchestration tools, which are more concerned with the business logic of an application, typically manage these interactions at a lower level of abstractions (*e.g.*, remote methods invocation).

4.2 Design support

This section presents the design support aspects that are common for both deployment and orchestration, *e.g.*, specification language, specification capabilities.

4.2.1 Specification language

To specify the orchestration and deployment of an IoT system, a language must be used.

Representation: The abstract syntax of a language defines its concepts and how they relate to each other. A concrete syntax, is concerned with the form (Moody, 2009) of a language and defines how abstract elements are realized in a concrete representation. A language may have textual and/or graphical syntaxes.

Language type: Languages for the deployment and orchestration of IoT can be a general purpose languages (GPL) but are typically designed specifically for that purpose (i.e., DSL) (Fowler, 2010). A DSL is either developed on top of an existing GPL (base elements of the GPL are extended) or from scratch and has its own custom concepts without explicit relationships to any existing language (Mernik et al., 2005).

Programming model: The selection of a programming model typically depends on the pragmatic of a tool. For instance, reactive and event-driven programming is often used to design IoT systems, flow-based programming is well suited for orchestrating data flows between IoT devices and services whilst component oriented programming is typically used to specify deployment models.

4.2.2 Specification capabilities

Here, we want to analyze how the DEPO4IOT approaches specify IoT applications and their corresponding deployment structure.

Application structure: To ensure separation of concerns, complex distributed systems are typically designed as a set of software components. A component is basically a unit of computation in an application, whereas interactions among components are represented by connectors (Medvidovic and Taylor, 2000). Components and connectors are used to describe the high-level structure of an application in terms of a component configuration. Practically, in this paper we call such components: *entities*. This is to avoid any confusion with the programming model used to implement them (*e.g.*, service, component). The application structure may also leverage predefined operators like Business Process Model and Notation (BPMN) operators for orchestration logic.

Deployment structure: Modelling a deployment structure typically overlaps with application structure because to specify the deployment of a system on a selected target infrastructure, its *entities* need to be allocated to IoT, edge or cloud resources. More precisely, what needs to be allocated on these resources are the implementations of those entities (Bergmayr et al., 2018). The notion of a deployable artifact supports exactly the reference between logical components and connectors to their implementations. For instance, a cloud application implemented in Java is possibly packaged into several archives, i.e., "JAR files". Those archives can be represented on the model level via artifacts. Allocating them to a cloud service, e.g., a compute service including a Java platform, should have the effect that the "JAR files" are physically allocated to the provisioned service (Bergmayr et al., 2018). Furthermore, we use the term dependencies to depict the fact that application entities (i) may need to interact with each other and (ii) may depend on other entities of specific platforms (Bergmayr et al., 2018). To actually ensure that connected cloud services can in fact interact with each other, properties related to networking concerns need often to be explicitly specified (Bergmayr et al., 2018).

4.3 Advanced support

Modern IoT systems must be trustworthy to really realize their values. We check whether the DEPO4IOT approaches address trustworthiness aspects (presented in Section 2.3), which often require advanced supports such as adaptation, monitoring, and shared access to resources.

Adaptation: (McKinley et al., 2004) define two main types of adaptation: (i) static adaptation and (ii) dynamic adaptation. Static adaptation typically happens at development time whilst dynamic adaptation happens at runtime. Orchestration and deployment tools can support these two types of adaptation for modifying: (i) the application itself (e.g., replacing one software component by another) and (ii) its infrastructure (e.g., bursting from one cloud to another).

Monitoring: Monitoring is a key activity to reason on the state of a system and for controlling and managing hardware as well as software infrastructures (Aceto et al., 2013). Monitoring probes are used to deliver information and indicators characterizing the system and the context in which it is running. The purpose of the monitoring can be multiple. In this study we identify the following: (i) measuring QoS of the system, (ii) capturing the status and health of a deployment, (iii) depicting the state of the environment, and (iv) observing the execution flow of the system, for instance for debugging purpose.

Shared access to resources (direct/indirect): Because IoT systems may involve actuators it is important to control the impact these actuators can have on the physical world and to manage conflicting actuation requests. More generally, this applies to the management of shared accesses to resources, which can be of two types: (i) direct concurrent access to resources (e.g., several entities accessing to the same service/actuator) or (ii) indirect shared access to a resource (e.g., actuators from different applications are modifying the temperature with possibly conflicting goals) (Ma et al., 2016).

5 RESULTS

Table 1 shows an overview of the primary DEPO4IOT studies. Based on the taxonomy, we have extracted and synthesized the data from the primary studies to answer the RQs of this work.

5.1 Answering RQ1

Answering RQ1.1: as we can see in Fig. 3, the earliest primary study was published in 2008, when IoT research was starting to emerge. There is a sharp rise in the number of DEPO4IOT publications in the last two years (2016, 2017), especially regarding the numbers of journal (J) and conference (C) papers (2016: 3J, 6C and 2017: 7J, 18C). This rise shows the crucial need of DEPO4IOT research and more attention to this research area has gained from the research community. We completed our search process in March 2018 and found five primary studies published in 2018 (3J, 2C).

IoT with its heterogeneous nature spans multiple relevant research domains among which we identified Software Engineering (SE), Cloud or Service-Oriented Architecture (SOA), Network, and recently specialized IoT research domain (Borgia et al., 2016). ***Answering RQ1.2:*** Fig. 4 shows the times that each research domain appears in the calls for papers in the publication venues where the primary DEPO4IOT studies published are quite close (SE: 22, IoT: 32, Clould/SOA: 26, Network: 18). Note that the publication venues can have not only one but several research domains in their calls for papers. These numbers do reflect the heterogeneous nature of IoT research, with IoT research domain is getting more visible. Due to the increasing popularity of the IoT research domain, and because the tools for the deployment and orchestration of application and services in the cloud have lately reached a high level of maturity, the focus for the research on deployment and orchestration approaches has moved toward the IoT and Edge spaces.

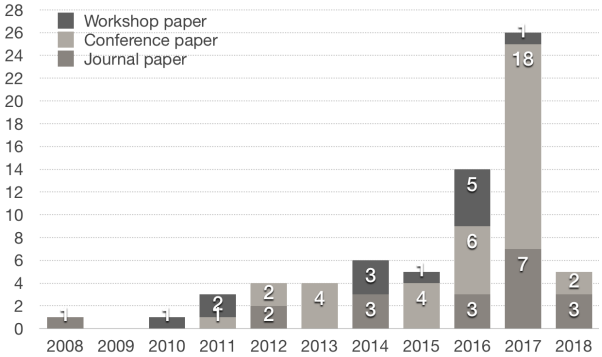

Figure 3: Publications per year, per venue type

Answering RQ1.3: By checking the affiliations of the authors, we can see in Fig. 5 that a majority of the authors publishing results on DEPO4IOT are academics (86%). The involvement of industry in this research is still very limited, which is understandable for a relatively new research area like IoT. ***Answering RQ2.5:*** Regarding the types of case studies used for evaluating the DEPO4IOT approaches, Fig. 6 also shows the similar dominance of academic approaches. We classify the case studies that

Table 1: Overview of the primary DEPO4IOT studies (sorted by year of publication)

#	*Title**	*Year*	v	f
1	Challenges and Solutions in Fog Computing Orchestration	2018	J	O
2	Deploying Edge Computing Nodes for Large-scale IoT: A Diversity Aware Approach	2018	J	D
3	Cloud-Fog Interoperability in IoT-enabled Healthcare Solutions	2018	C	D
4	A visual programming framework for distributed Internet of Things centric complex event processing	2018	J	B
5	Enhancing Middleware-based IoT Applications through Run-Time Pluggable QoS Management Mechanism	2018	C	D
6	A Dynamic Module Deployment Framework for M2M Platforms	2017	C	D
7	A Middleware for Mobile Edge Computing	2017	J	B
8	A service orchestration architecture for Fog-enabled infrastructures	2017	C	O
9	Distributed Orchestration in Large-Scale IoT Systems	2017	C	O
10	Internet of Things: From Small- to Large-Scale Orchestration	2017	C	O
11	QoS-Aware Deployment of IoT Applications Through the Fog	2017	J	D
12	Service Orchestration in Fog Environments	2017	C	O
13	Towards Container Orchestration in Fog Computing Infrastructures	2017	C	O
14	A Framework based on SDN and Containers for Dynamic Service Chains on IoT Gateways	2017	W	D
15	A framework for MDE of IoT-Based Manufacturing CPS	2017	C	O
16	A Novel Service-Oriented Platform for the Internet of Things	2017	C	O
17	Design and Implementation of a Message-Service Oriented Middleware for Fog of Things Platforms	2017	C	O
18	Empowering End Users to Customize their Smart Environments	2017	J	O
19	Feasibility of Fog Computing Deployment based on Docker Containerization over RaspberryPi	2017	C	B
20	Semantics Based Service Orchestration in IoT	2017	C	O
21	An Object-Oriented Model for Object Orchestration	2017	C	B
22	A TOSCA-based Programming Model for Interacting Components of Automatically Deployed Cloud and IoT Applications	2017	C	B
23	An edge-based platform for dynamic Smart City applications	2017	J	D
24	Calvin Constrained: A Framework for IoT Applications in Heterogeneous Environments	2017	C	B
25	InterCloud Communication Through Gatekeepers to Support IoT Service Interaction in the Arrowhead Framework	2017	J	O
26	Internet of things out of the box Using TOSCA for automating the deployment of IoT environments	2017	C	B
27	Platform-as-a-service gateway for the Fog of Things	2017	J	B
28	Runtime deployment and management of CoAP resources for the internet of things	2017	J	D
29	Composing Continuous Services in a CoAP-based IoT	2017	C	B
30	Niflheim: An end-to-end middleware for applications on a multi-tier IoT infrastructure	2017	C	B
31	Foggy- A Framework for Continuous Automated IoT Application Deployment in Fog Computing	2017	C	D
32	A Web of Things Based Device-Adaptive Service Composition Framework	2016	C	O
33	Application Orchestration in Mobile Edge Cloud: Placing of IoT Applications to the Edge	2016	W	O
34	Optimizing Elastic IoT Application Deployments	2016	J	D
35	FRED- A Hosted Data Flow Platform for the IoT built using NodeRED	2016	W	B
36	Incremental deployment and migration of geo-distributed situation awareness applications in the fog	2016	C	D
37	On Building Smart City IoT Applications- a Coordination-based Perspective	2016	W	B
38	Orchestrating the Internet of Things Dynamically	2016	W	B
39	SoIoT: Toward A User-Centric IoT-Based Service Framework	2016	J	O
40	A Container-based Edge Cloud PaaS Architecture-based on Raspberry Pi Clusters	2016	C	B
41	Automated Deployment of SmartX IoT-Cloud Services based on Continuous Integration	2016	C	D
42	Cloud4IoT: A Heterogeneous, Distributed and Autonomic Cloud Platform for the IoT	2016	C	B
43	Reliable services composition method for the internet of thing using directed service-object graph deployment scheme	2016	C	O
44	Integration of Heterogeneous Services and Things into Choreographies	2016	W	O
45	A Scalable Framework for Provisioning Large-Scale IoT Deployments	2016	J	D
46	A Data-Centric Framework for Development and Deployment of Internet of Things Applications In Clouds	2015	C	D
47	Towards a Semantic Model for Automated Deployment of IoT Services across Platforms	2015	C	D
48	A component based approach for the Web of Things	2015	W	O
49	A Generic Service Oriented Software Platform to Design Ambient Intelligent Systems	2015	C	O
50	Developing IoT Applications in the Fog: a Distributed Dataflow Approach	2015	C	B
51	A Full End-to-End Platform as a Service for Smart City Applications	2014	W	B
52	A Novel Deployment Scheme for Green Internet of Things	2014	J	D
53	glue.things - a Mashup Platform for wiring the Internet of Things with the Internet of Services	2014	W	B
54	Toward a Distributed Data Flow Platform for the Web of Things	2014	W	O
55	BeC3: Behaviour Crowd Centric Composition for IoT applications	2014	J	B
56	Dioptase: a distributed data streaming middleware for the future web of things	2014	J	B
57	Application deployment for IoT: An infrastructure approach	2013	C	B
58	Orchestration in distributed web-of-objects for creation of user-centered iot service capability	2013	C	O
59	Towards Automated IoT Application Deployment by a Cloud-Based Approach	2013	C	D
60	Mobile Fog: A Programming Model for Large-Scale Applications on the Internet of Things	2013	C	D
61	Gateway as a service- A cloud computing framework for web of things	2012	C	O
62	Behaviour-Aware Compositions of Things	2012	C	O
63	Knowledge-Aware and Service-Oriented Middleware for deploying	2012	J	B
64	Mashing up the Internet of Things: A framework for smart environments	2012	J	O
65	D-LITE: Distributed logic for internet of things services	2011	C	B
66	Adaptable Service Composition for Very-Large-Scale IoT Systems	2011	W	O
67	INOX: A managed service platform for interconnected smart objects	2011	W	B
68	Connecting Smart Things through Web Services Orchestrations	2010	W	O
69	COSMOS: a middleware for integrated data processing over heterogeneous sensor networks	2008	J	O

PV = Short name of publication venue.
[v] Venue type: J = Journal (19), C = Conference (37), W = Workshop (13).
[f] Focus: D = Deployment, O = Orchestration, B = Both Deployment and Orchestration.
* The titles are clickable to link to the corresponding publications

are not from industry as academic ones, *e.g.*, motivational examples, prototypes or simulations developed by researchers for discussing or evaluating their DEPO4IOT approaches. Nearly one tenth of the studies do not really provide evaluation details (not available, N/A) because of the early stage of their work, *e.g.*, reported in short papers or new ideas papers. But, the number of industrial case studies or empirical studies (with industry) account for 13% in total, which is still encouraging. We would call for more collaboration between academia and industry for more practical DEPO4IOT research in particular, but also IoT research in general.

5.2 Answering RQ2: Deployment & Orchestration support

Answering RQ2.1 and RQ2.2: Fig. 7 shows the distribution of primary studies based on the focus:

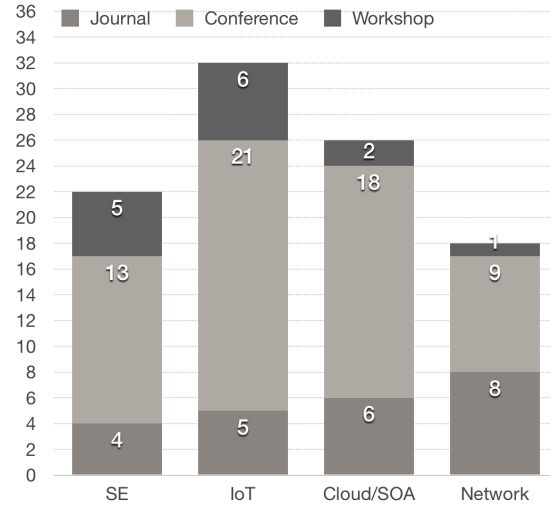

Figure 4: Research topics per publication venue

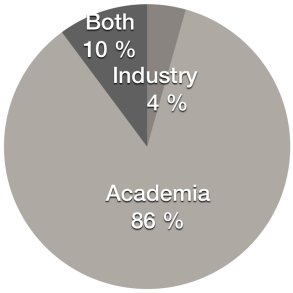

Figure 5: The affiliations of authors

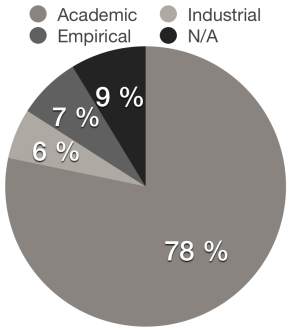

Figure 6: Evaluation case studies

orchestration, or deployment, or both orchestration and deployment. We can see that the orchestration-focused studies are nearly double the deployment-focused studies. The main reason could be that the orchestration-focused studies are around IoT data mash-up at cloud or edge level, which we show later in Fig. 8. It is understandable that the first IoT research focus is more on building IoT applications (orchestration) before deploying them. These studies do not technically contribute to low-level orchestration involving IoT devices or gateways but rather make assumptions on low-level IoT infrastructure. About one-third (37%) of the approaches support both deployment and orchestration. Note that the degrees of support for deployment and orchestration are not at the same level. In other words, approaches that support deployment somehow also support orchestration (as part of the deployment specifications overlap with orchestration specifications). However, the orchestration support in these cases is often at a higher level of abstraction than the orchestration-focused approaches that offer specific support for orchestration (*i.e.*, logical port vs. method).

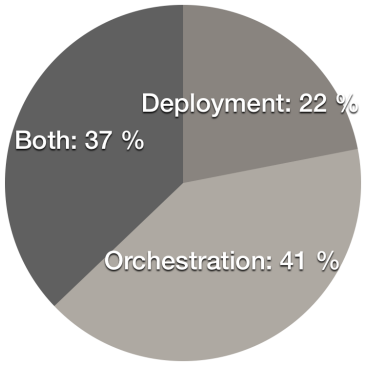

Figure 7: Main focus of the primary DEPO4IOT studies

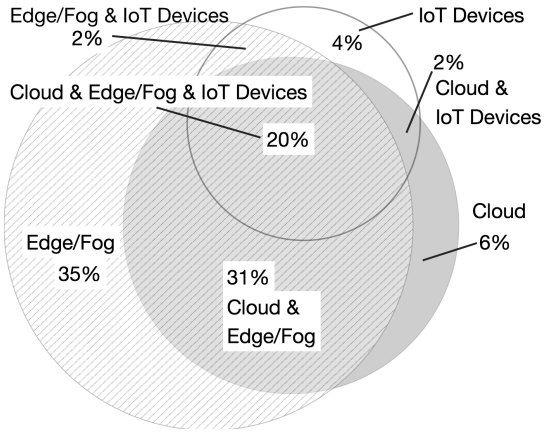

Figure 8: Target infrastructure

Fig. 8 shows how the primary DEPO4IOT studies support for different layers of IoT infrastructure: cloud, edge, or IoT devices. Most studies (71%) discuss about mashing up data streaming from IoT devices at cloud (7%) or edge/fog (32%) or both (32%). But, few studies (29%) really support orchestrating and/or deploying software on IoT devices. We would argue that deployment and orchestration at IoT devices are the most challenging research problems in DEPO4IOT because of the diversity of IoT devices, their networking protocols and connectivity issues, and their different computing resource constraints. To really support for modern real IoT systems in which trustworthy aspects are crucial, DEPO4IOT studies must advance to the technical details of edges and IoT devices. Even among the 29% mentioned above, we find very limited support for modern IoT systems as we discuss in the following paragraphs.

Looking closer into the studies that support deployment (59%, Fig. 7), we find that nearly two-third (64%, Fig. 9) of them have declarative deployment type vs. one-third have imperative type. We would

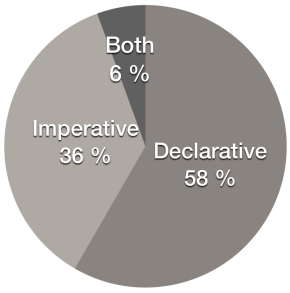

Figure 9: Deployment engine

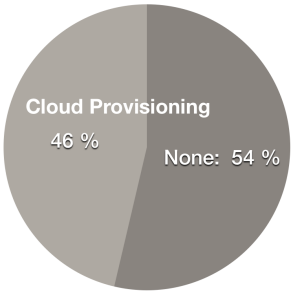

Figure 10: Cloud provisioning

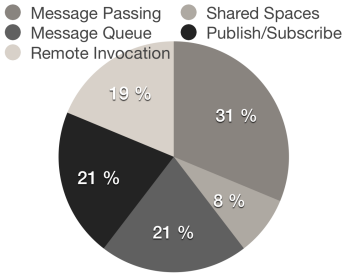

Figure 11: Orchestration communication types

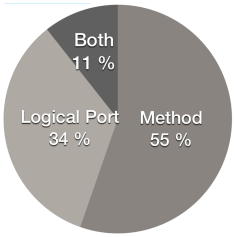

Figure 12: Integration level

argue this is because imperative approaches are typically more complicated to specify and reuse (as they require the specification of a deployment plan). On the other hand, they give full control over the deployment process, thus allowing its optimization. Very few approaches (6%) provide users with the ability to select between both approaches. We would also argue that hybrid approaches could be beneficial. For instance, a deployment engine could first derives a tentative deployment plan from a deployment configuration describing the desired system state. Then, if necessary, the user or a reasoning engine could use an imperative approach to adjust this plan before the actual deployment. Also among 59% of the studies that support deployment, less than half (46%) support the provisioning of cloud resources before deployment (Fig. 10). This means only one-third (59% times 46%) of the total primary studies support cloud resources provisioning, which is an important feature in any advanced DEPO4IOT approach. This illustrates the need for further integration of the works from the cloud and IoT domains.

Among the studies that support orchestration (78%, Fig. 7), we find that the communication types in orchestrating IoT components are diverse (Fig. 11). However, we observe that approaches offering the best decoupling in time and space (Eugster et al., 2003) (between subscribers and publisher), *i.e.*, message passing, message queues and publish/subscribe, are largely adopted (73% in total). The integration level at methods (66%, Fig. 12) for orchestration is more common than at logical port (34%). Only 11% support both levels, which is often needed for orchestrating more complex IoT systems.

5.3 Answering RQ2: Design support

This section gives ***more answers to the RQ2.1 and RQ2.2***, regarding the design support aspects of orchestration and deployment. Fig. 13 shows that the most common term to depict deployment and orchestration entities is *service*, which is significantly more often than *component* and *node*. We find that most orchestration approaches have used services for data mash-up. *Node* term seems getting more common in supporting both deployment and orchestration (*e.g.*, in NodeRED-based approaches). The total number of primary studies that provide technical information about connectors is 43, which is two-third of the DEPO4IOT approaches. High-level operators (like BPMN operators) for supporting deployment and orchestration are not common because very few have been mentioned in the approaches.

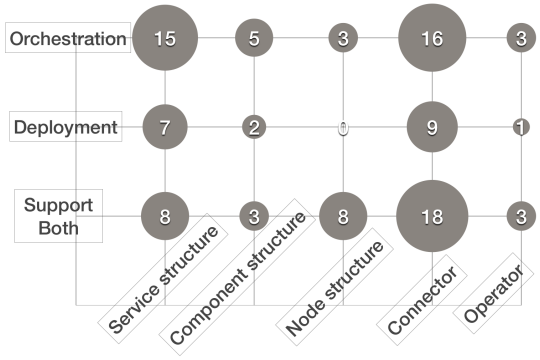

Figure 13: Application structure

Any thorough DEPO4IOT approaches, especially for deployment and/or orchestration at IoT devices level, should have provided technical information about bootstrap and network specification. But, Fig. 14 shows that only less than one-third (32%) of the deployment approaches have provided any such information. This could be interpreted that most current approaches do not address the technical details of deployment and make assumptions on the support of underlying operating systems or execution engines. We also find in Fig. 14 only about half having either bootstrap (52%) or network specification (55%) with deployment. A thorough deployment approach should have included network specification(s) to support for low-level deployment on IoT devices, where diverse network topologies and protocols must be considered.

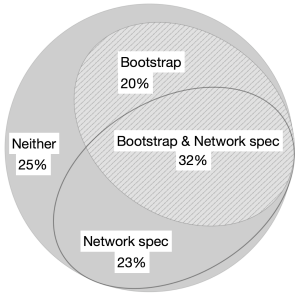

Figure 14: Bootstrap and network specification provided with deployment?

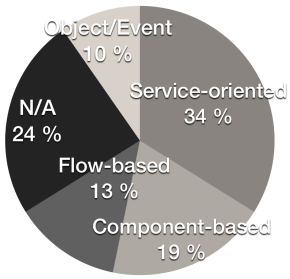

Figure 15: Programming model

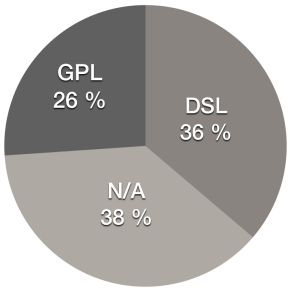

Figure 16: Language types

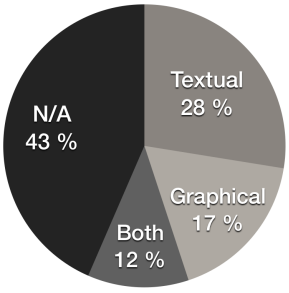

Figure 17: Representation kinds

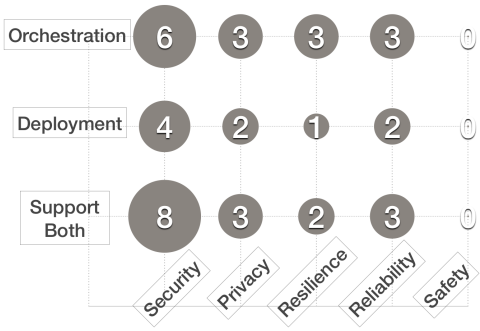

Figure 18: Trustworthiness aspects

Figure 19: Advanced supports

Aligned with the popular of "service" discussed in the previous paragraphs, service-oriented model is the most common (34%, Fig. 15) programming model in the DEPO4IOT approaches, followed by component-based (19%), and flow-based (13%). These models can provide a modular, loosely-coupled specification of the orchestration and its deployment so that the modules can be seamlessly substituted and reused.

The language support is important to make DEPO4IOT approaches more practical, but more than one-third of the approaches do not provide language support for deployment and orchestration (N/A in Fig. 16). Using DSL seems to be more popular than GPL because DSL could be more expressive in specifying DEPO4IOT aspects. Fig. 17 shows that textual is the most popular form of language support (28%), compared to graphical (17%). Some approaches (12%) do have both graphical and textual formats.

5.4 Answering RQ2: Trustworthiness & advanced supports

Answering RQ2.3 and RQ2.4: Trustworthiness aspects must be supported in the orchestration and deployment of modern IoT systems. However, Fig. 18 shows that very few primary DEPO4IOT studies address trustworthy aspects: security (18 out of 69 studies), privacy (8), resilience (6), reliability (8). Among the trustworthiness aspects, security is the most addressed one in the primary DEPO4IOT studies. Even so, most of 18 DEPO4IOT studies addressing security only briefly mention security aspects in their DEPO4IOT approaches. Studies that address security in more details are very rare (*e.g.*, the study #24 in Table 1). None addresses safety, which should be also a crucial property of critical IoT systems. This relates to the Shared access to resources (including access to actuators), which is a rare feature in the existing DEPO4IOT approaches (*e.g.*, in the study #49). We believe a reason for that is that IoT system innovations have until now mainly been concerned with sensors, device management and connectivity, with the mission to gather data for processing and analysis in the cloud in order to aggregate information and knowledge[8]. Management of actuators has thus only recently appeared as a research priority in the community. In general, Fig. 19 shows that few primary DEPO4IOT studies provide advanced supports such as monitoring, runtime adaptation, shared access. It is worth noting that it is often the same approaches that support adaptation and monitoring. We believe this is due to the fact that both are complementary.

[8]IEC white paper entitled "IoT 2020: Smart and secure IoT platform". http://www.iec.ch/whitepaper/pdf/iecWP-IoT2020-LR.pdf

Indeed, before triggering an adaptation it is important to observe and understand the status of the system.

5.5 Answering RQ3

This section gives ***our answers to the RQ3.1 and RQ3.2*** that are supported by the findings presented above. Even though there is a big jump in the number of primary DEPO4IOT studies recently, our analyses show that DEPO4IOT research is still in its infancy. Thus, there are fundamental technical gaps and open issues to be tackled.

We found that the number of orchestration approaches, often for IoT data mash-up, is nearly double the number of deployment approaches. The orchestration approaches for IoT data mash-up such as the studies numbered 10, 20, 33, 62, 64 in Table 1 often make assumptions of readily deployed IoT systems in operation. IoT deployment approaches should receive more attention. Without proper deployment approaches to deploy IoT systems, orchestration approaches cannot thrive. Besides, it is worth to note that most of the primary DEPO4IOT approaches do not really support deployment and/or orchestration at IoT devices, *e.g.*, without bootstrap or network specification. Only by going into low-level details, IoT engineering can really control the whole chain of IoT software deployed from cloud until IoT devices. In this way, it would be more likely that the trustworthy aspects and advanced supports can be addressed more systematically and efficiently.

They are also immature in addressing trustworthy aspects and advanced supports. The number of existing DEPO4IOT studies addressing trustworthy aspects is very small. Even among those studies, we have not really found any that explicitly and systematically considered trustworthy aspects. The same observation applies to the advanced support of DEPO4IOT studies, *i.e.*, regarding monitoring, adaptation, and shared access to resources. We would propose that new DEPO4IOT research should address more thoroughly and systematically the trustworthy aspects and advanced supports, which are crucial for modern IoT systems.

Finally, the dominance of academia-only in DEPO4IOT research suggests that there should be more collaboration between academia and industry to make DEPO4IOT approaches more practical and closer to the needs in industry.

6 Related work

IoT has emerged as an important area of research and development in the recent years. There have been efforts to review the state of the art in different aspects of IoT. Some surveys have addressed IoT middleware (Razzaque et al., 2016; Ngu et al., 2017), IoT platforms (Mineraud et al., 2016), and IoT architectural concerns (Gill et al., 2017) but none has systematically, specifically investigated deployment and orchestration approaches for IoT. Note that we have clearly defined the scope of our SMS, which only considered peer-reviewed publications, not white papers from industry. Thus, our SMS reports the state of the art in DEPO4IOT research, not including the state of practice in industry.

From the software architecture view, an IoT middleware provides a layer between application software and system software. DEPO4IOT approaches are not necessary about middleware because they are more vertical along the IoT engineering lifecycle from development to operation of IoT systems. Research on IoT middleware highly overlaps with DEPO4IOT because DEPO4IOT approaches often leverage middlewares for integrating heterogeneous computing and communications devices, and supporting interoperability within the diverse applications and services running on these devices. In (Razzaque et al., 2016) and (Ngu et al., 2017), the authors conducted two different surveys of the existing middlewares to classify them and find out the main challenges. The results of these studies not only address the functional aspects of IoT middleware but also some quality aspects such as security, adaptation that we also considered in our work. But, these two studies are not systematic studies.

(Mineraud et al., 2016) provide a gap analysis on the well-known IoT platforms. In particular, the authors identify gaps related security (fine grained access control), cross-platform and cross-layer DSL to reduce threats on privacy and security. Their work is not a systematic study and mainly focuses on platforms maturity and usability. (Gill et al., 2017) conducted a systematic study on the key IoT architectural concerns. It highlights a set of challenges that is a combination of technical, human, financial and ethical aspects. Their work is complementary to our work reported in this paper as it does not focus on deployment, orchestration and trustworthiness. However, our work and (Gill et al., 2017) share some common findings, *e.g.*, the lack of support for IoT security, and the need for runtime adaptation of IoT systems.

7 CONCLUSIONS

Deployment and orchestration approaches for IoT should be advanced enough to support distributed processing and coordinated behavior across IoT, edge and cloud infrastructures. In this paper, we have examined the research landscape of deployment and orchestration approaches for IoT, by conducting a systematic mapping study. After systematically identifying and reviewing 69 primary studies out of thousands relevant papers in this field, we have found out that 1) there is a sharp rise in the number of publications addressing this field in the two recent years; 2) however, there are still different gaps that the current approaches seem to be immature to address such as the real, low-level technical details of deployment and/or orchestration at IoT devices level; 3) new deployment and/or orchestration approaches should also focus on addressing the trustworthy aspects and advanced supports for modern IoT systems. To make the IoT deployment and orchestration approaches more practical, there should be more and more research collaborations between academia and industry. We will address these open issues in our current research projects, especially for the trustworthiness of IoT.

REFERENCES

A. Metzger (Ed.) (2015). Cyber physical systems: Opportunities and challenges for software, services, cloud and data.

Aceto, G., Botta, A., De Donato, W., and Pescapè, A. (2013). Cloud monitoring: A survey. *Computer Networks*, 57(9):2093–2115.

Arcangeli, J.-P., Boujbel, R., and Leriche, S. (2015). Automatic deployment of distributed software systems: Definitions and state of the art. *Journal of Systems and Software*, 103:198–218.

Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., et al. (2010). A view of cloud computing. *Communications of the ACM*, 53(4):50–58.

Bergmayr, A., Breitenbücher, U., Ferry, N., Rossini, A., Solberg, A., Wimmer, M., Kappel, G., and Leymann, F. (2018). A systematic review of cloud modeling languages. *ACM Comput. Surv.*, 51(1):22:1–22:38.

Binz, T., Breitenbücher, U., Haupt, F., Kopp, O., Leymann, F., Nowak, A., and Wagner, S. (2013). Opentosca–a runtime for tosca-based cloud applications. In *International Conference on Service-Oriented Computing*, pages 692–695. Springer.

Bonomi, F., Milito, R., Zhu, J., and Addepalli, S. (2012). Fog computing and its role in the internet of things. In *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, pages 13–16. ACM.

Borgia, E., Gomes, D. G., Lagesse, B., Lea, R. J., and Puccinelli, D. (2016). Special issue on "internet of things: Research challenges and solutions". *Computer Communications*, 89:1–4.

Carzaniga, A., Fuggetta, A., Hall, R. S., Heimbigner, D., Van Der Hoek, A., and Wolf, A. L. (1998). A characterization framework for software deployment technologies. Technical report, Colorado State Univ Fort Collins Dept Of Computer Science.

Dearie, A. (2007). Software deployment, past, present and future. In *Future of Software Engineering, 2007. FOSE'07*, pages 269–284. IEEE.

Eugster, P. T., Felber, P. A., Guerraoui, R., and Kermarrec, A.-M. (2003). The many faces of publish/subscribe. *ACM computing surveys (CSUR)*, 35(2):114–131.

Fowler, M. (2010). *Domain-specific languages*. Pearson Education.

Gill, A. Q., Behbood, V., Ramadan-Jradi, R., and Beydoun, G. (2017). Iot architectural concerns: a systematic review. In *Proceedings of the Second International Conference on Internet of things and Cloud Computing*, page 117. ACM.

Griffor, E. R., Greer, C., Wollman, D. A., and Burns, M. J. (2017). Framework for cyber-physical systems: Volume 1, overview. Technical report.

IEC, ISO (2011). Information technology-security techniques-privacy framework. *International Standard No. ISO/IEC*.

IEC, ISO (2012). Information technology-security techniques-information security management systems-overview and vocabulary. *International Standard No. ISO/IEC*, 27000:32.

IEEE Standards Association et al. (2016). P2413-standard for an architectural framework for the internet of things (iot). *Institute of Electrical and Electronics Engineers, New York*.

Kitchenham, B. (2004). Procedures for performing systematic reviews. *Keele, UK, Keele University*, 33(2004):1–26.

Ma, M., Preum, S. M., Tarneberg, W., Ahmed, M., Ruiters, M., and Stankovic, J. (2016). Detection of runtime conflicts among services in smart cities. In *Smart Computing (SMARTCOMP), 2016 IEEE International Conference on*, pages 1–10. IEEE.

McKinley, P. K., Sadjadi, S. M., Kasten, E. P., and Cheng, B. H. (2004). A taxonomy of compositional adaptation. *Rapport Technique*.

Medvidovic, N. and Taylor, R. N. (2000). A classification and comparison framework for software architecture description languages. *IEEE Transactions on software engineering*, 26(1):70–93.

Mell, P. and Grance, T. (2001). The NIST Definition of Cloud Computing. Special Publication 800-145, National Institute of Standards and Technology.

Mernik, M., Heering, J., and Sloane, A. M. (2005). When and how to develop domain-specific languages. *ACM computing surveys (CSUR)*.

Mineraud, J., Mazhelis, O., Su, X., and Tarkoma, S. (2016). A gap analysis of internet-of-things platforms. *Computer Communications*, 89.

Moody, D. (2009). The "physics" of notations: toward a scientific basis for constructing visual notations in software engineering. *IEEE Transactions on Software Engineering*, 35(6):756–779.

Ngu, A. H., Gutierrez, M., Metsis, V., Nepal, S., and Sheng, Q. Z. (2017). Iot middleware: A survey on issues and enabling technologies. *IEEE Internet of Things Journal*, 4(1):1–20.

Petersen, K., Vakkalanka, S., and Kuzniarz, L. (2015). Guidelines for conducting systematic mapping studies in software engineering: An update. *Information and Software Technology*, 64:1–18.

Razzaque, M. A., Milojevic-Jevric, M., Palade, A., and Clarke, S. (2016). Middleware for internet of things: a survey. *IEEE Internet of Things Journal*, 3(1):70–95.

Tran, N. K., Sheng, Q. Z., Babar, M. A., and Yao, L. (2017). Searching the web of things: state of the art, challenges, and solutions. *ACM Computing Surveys (CSUR)*, 50(4):55.