Cloud Container Technologies: A State-of-the-Art Review

Claus Pahl, Antonio Brogi, Jacopo Soldani, and Pooyan Jamshidi

Abstract—Containers as a lightweight technology to virtualise applications have recently been successful, particularly to manage applications in the cloud. Often, the management of clusters of containers becomes essential and the orchestration of the construction and deployment becomes a central problem. This emerging topic has been taken up by researchers, but there is currently no secondary study to consolidate this research. We aim to identify, taxonomically classify and systematically compare the existing research body on containers and their orchestration and specifically the application of this technology in the cloud. We have conducted a systematic mapping study of 46 selected studies. We classified and compared the selected studies based on a characterisation framework. This results in a discussion of agreed and emerging concerns in the container orchestration space, positioning it within the cloud context, but also moving it closer to current concerns in cloud platforms, microservices and continuous development.

Index Terms—Cloud, container, container technologies, orchestration, cluster, systematic review

1 Introduction

Containerisation is a technology to virtualise applications in a lightweight way that has resulted in a significant uptake in cloud applications management. How to orchestrate the construction and deployment of containers individually and in clusters has become a central problem [13].

There has not been a secondary study of research on container technologies in the cloud that would allow to assess the maturity in general and identify trends, research gaps and future directions. Given the growing interest in containers, their management and orchestration in cloud, there is a need to explore current research. Secondary studies identify, classify and synthesise a comparative overview of state-of-the-research and enable an assessment of ongoing work [7], [15]. We opt for a systematic mapping study (SMS) as it is more suitable in mapping out and structuring new areas of investigation.

We identify, taxonomically classify and systematically compare the existing research body on container technologies and its application in the cloud, aiming to extract a better understanding of Platform-as-a-Service (PaaS) as middleware built on containers for application packaging and as a deployment infrastructure. We have conducted a systematic mapping study of 46 selected studies (Table 2), spanning over a decade from 2007 onwards. We classified and compared the selected studies based on a characterisation framework.

- *C. Pahl is with the Faculty of Computer Science, Free University of Bozen-Bolzano, Bolzano 39100, Italy. E-mail: claus.pahl@unibz.it.*
- *A. Brogi and J. Soldani are with the Department of Computer Science, University of Pisa, Pisa 56126, Italy. E-mail: {brogi, soldani}@di.unipi.it.*
- *P. Jamshidi is with the School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213. E-mail: pjamshid@cs.cmu.edu.*

Manuscript received 25 Oct. 2016; revised 10 Mar. 2017; accepted 22 Apr. 2017. Date of publication 9 May 2017; date of current version 4 Sept. 2019. (Corresponding author: Claus Pahl.)
Recommended for acceptance by P. Balaji.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TCC.2017.2702586

Our mapping study resulted in a knowledge base of current research approaches, methods, techniques, best practices and experiences used in cloud architecture, with a particular attention to cloud application development and management. Our study revealed that container technologies research is still in a formative stage. More experimental and empirical evaluation of benefits is needed. Our study also showed a lack of tool support to automate and facilitate container management and orchestration, specifically in clustered cloud architectures.

The results of our mapping study show growing interests and usage of container-based technologies (such as LXC or Docker) as lightweight virtualisation solutions at Infrastructure-as-a-Service (IaaS) level, and as application management solutions at PaaS level. We can observe that containers positively impact on both development and deployment aspects. For instance, architecting in the cloud moves towards DevOps-based approaches, supporting a continuous development and deployment pipeline taking into account cloud-native architecture solutions based on containers and their orchestration (Brunnert et al., 2015). The results show that containers can support continuous development in the cloud based on cloud-native platform services for development and deployment, but do require advanced orchestration support. Container-based orchestration techniques hence emerge as a mechanism to orchestrate computation in cloud-based, clustered environments. The results of our study show that such techniques are seen to balance the need of technical quality management, e.g., optimised resource utilisation and performances, which is a cost factor in the cloud (due to its utility pricing principle).

Our systematic mapping study aims to benefit, first, researchers in software engineering, distributed systems and cloud computing, who need an identification of relevant studies. A systematic presentation of research provides a body of knowledge to develop theory and solutions, analyse research implications and establish future dimensions. It

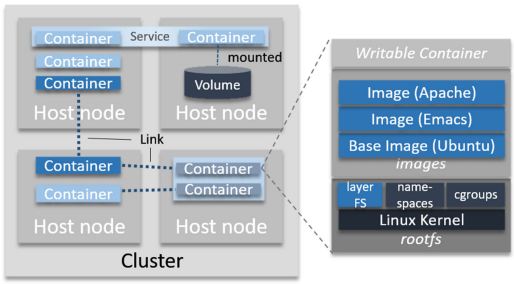

Fig. 1. Container cluster architectures.

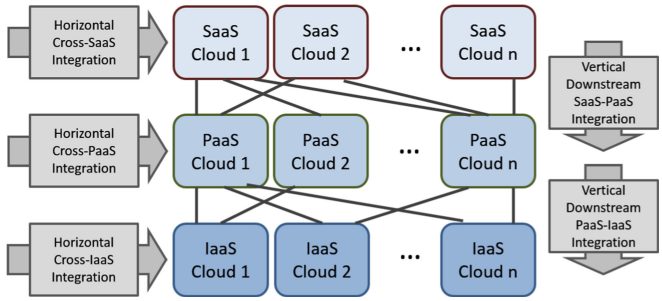

Fig. 2. Cloud reference architecture model.

also benefits practitioners interested in understanding the available methods, techniques and tools as well as their constraints and maturity level.

This paper is structured as follows. Section 2 describes background and related research to position this work. Section 3 explains the research methodology, research questions and scope. Section 4 provides a characterisation framework for cloud container orchestration. Section 5 presents the results of the mapping study, followed by an analysis of its limitations. Section 6 discusses findings, implications and trends.

2 Container Architectures and Their Management

The cloud uses virtualisation techniques to achieve elasticity of large-scale shared resources [12]. Virtual machines (VMs) are typically the backbone at the infrastructure layer. Containerisation in contrast allows a lightweight virtualisation through the bespoke construction of containers as application packages from individual images (generally retrieved from an image repository) that consume less resources and time. They also support a more interoperable application packaging needed for portable, interoperable software applications in the cloud [13]. Containerisation is based on the capability to develop, test and deploy applications to a large number of servers and also to interconnect these containers. Containers address consequently concerns at the cloud PaaS level. Given the overall importance of the cloud, a consolidating view on current activities is important.

2.1 Container Technology Principles

A container holds packaged self-contained, ready-to-deploy parts of applications and, if necessary, middleware and business logic (in binaries and libraries) to run the applications. Tools like Docker are built around container engines where containers act as portable means to package applications. This results in the need to manage dependencies between containers in multi-tier applications. An orchestration plan can describe components, their dependencies and their lifecycle in a layered plan. A PaaS cloud can then execute the workflows from the plan through agents (like a container engine). PaaS clouds can consequently support the deployment of applications from containers. Orchestration subsumes here their coordinated construction, deployment and ongoing management [10].

Many container solutions are based on Linux LXC techniques. Recent Linux distributions-part of the Linux container project LXC-provide kernel mechanisms such as *namespaces* and *cgroups* to isolate processes on a shared operating system [S5]. Docker is the most popular container solution at the moment and shall be used to illustrate containerisation. A Docker image is made up of file systems layered over each other, similar to the Linux virtualisation stack, using the LXC mechanisms, see Fig. 1 (right). Docker uses a union mount to add a writable file system on top of the read-only file system. This allows multiple read-only file systems to be stacked on top of each other. This property can be used to create new images by building on top of base images. Only the top layer is writable, which is the container itself.

Containerisation facilitates the step from single applications in containers to clusters of container hosts that can run containerised applications across cluster hosts [S9]. The latter benefits from the built-in interoperability of containers. Individual container hosts are grouped into interconnected clusters, illustrated in Fig. 1 (left). Each cluster consists of several (host) nodes. Application services are logical groups of containers from the same image. Application services allow scaling an application across different host nodes. Volumes are mechanisms used for applications that require data persistence. Containers can mount these volumes for storage. Links allow two or more containers to connect and communicate. The set-up and management of these container clusters requires orchestration support for inter-container communication, links and service assemblies [13].

2.2 Cloud-Based Container Architectures

Container orchestration deals not only with turning applications on or off (i.e., start or stop containers) and moving them among servers. We define orchestration as constructing and continuously managing possibly distributed clusters of container-based software applications. Container orchestration allows users to define how to coordinate the containers in the cloud when a multi-container application is deployed. Container orchestration defines not only the initial deployment of containers, but also the management of the multi-containers as a single entity. It takes care of availability, scaling and networking of containers. Essentially cloud-based container construction is a form of orchestration within the distributed cloud environment. The cloud can be seen as a distributed and tiered architecture, see Fig. 2, with core infrastructure, platform and software application tiers distributed across multi-cloud environments [1]. Container technologies can help. As such, container technologies will play a central role in the future of application management, in particular in the cloud PaaS context.

Recen tly popular microservice-based architectures can be realised in this cloud framework through containers [8],

[9]. Given this change in architecting, a secondary study can help practitioners for their decision making in terms of the correct technology choice.

2.3 State-of-the-Art

The mechanism we use to review cloud container techniques is that of a systematic mapping study. Reviews can be distinguished into two forms. Systematic Literature Reviews (SLR) suit summative analyses of mature fields, based on possibly larger bodies of literature. Systematic Mapping Studies are suitable to determine the structure of the type of research reports, visual categorisation, useful if there is a lack of high-quality primary studies. SMSs are typically less detailed, but they more appropriate for our purposes.

As part of our paper selection process, we extracted six review papers that are related to our aim (see Table 2). Five out of these six qualify as technology reviews, i.e., they overview and assess container technologies. Study [S3] covers virtualisation basics and container construction/management. The focus is more on deployment than development. Study [S4] clearly addresses virtualisation basics only, but from deployment and development perspectives. Study [S6] is then more comprehensive, including clusters and both deployment and development perspectives. Study [S9] is similar to [S6], but has less quality management concerns covered. Study [S19] focusses like [S4] on virtualisation basics, but specifically on performance in HPC and computation/storage intensive applications. Slightly different in the approach is study [S17], which is more organised around research coverage rather than only technology. However, a systematic coverage of literature (like the one we propose in this paper) is missing.

Review Step	Activity
plan	identify need, specify reseach questions, define protocol
conduct	select primary studies, extract/synthesise data
document	document observations, analyse threats, report

SLRs and SMSs entail to identify, classify and compare existing evidence on the use of container technologies specifically in cloud environments through a characterisation framework. As highlighted above, some technology reviews exist, but these concentrate on technology and do not capture research efforts and directions systematically.

3 Research Methodology

3.1 A Systematic Mapping Process

SMSs reduce bias through a rigorous sequence of methodological steps to search and classify literature. They rely on well-defined and evaluated review protocols to extract, analyse and document results. We follow the process presented in [15] with a three-step review that includes planning, conducting and documenting.

The review is complemented by an evaluation of each step's outcome. Furthermore, we provide an additional characterisation framework for the study context. We have adapted and applied a systematic mapping to cloud technology in a study focusing on container orchestration. The essential process steps of our systematic mapping study are definition of research questions, conducting the search for relevant papers, screening of papers, keywording of abstracts and data extraction and mapping. Each process step has an outcome, the final outcome of the process being the systematic map:

Now, the individual steps of the three-step process above will be outlined. Based on the objectives, we first specify the research questions and the review scope in order to formulate search strings for literature extraction.

Process Steps	Outcomes	Section
Definition of Research Question	Review Scope	3.2
Conduct Search	All Papers	3.3
Screening of Papers	Relevant Papers	3.4
Keywording using Title and Abstract	Classification Scheme	3.5
Data Extraction & Mapping Process	Systematic Map	3.6

PICO concern	Explanation
Population	RQ1: Practical motivation RQ2: Structure and architecture aspects RQ3: Management methods and techniques RQ4: Research challenges and future dimensions
Intervention	characterise, internal/external validation; extract data; synthesis
Comparison	compare by mapping primary studies to characterisation framework
Outcome	a characterisation framework

Identify the Scope of our SMS. We already discussed the need for a SMS. We can also clarify the general goal and scope of the study using the Population, Intervention, Comparison, Outcome (PICO) criteria [7]:

Define and Evaluate Review Protocol. We developed a protocol based on [15] and on our experience with SLRs [4], [5]. Conducting the review starts with the study selection and results in extracted data and synthesised information. We specifically focus on orchestration to capture the trend towards distributed container architectures from a research perspective (using orchestration as a broad inclusive term).

3.2 Definition of Research Questions (Review Scope)

As the next activity, we *define the research questions* to help shaping the review protocol, see Table 1. The main goal of a systematic mapping study is to provide an overview of a research area and to identify the quantity and type of research and results available within it. We can map the frequencies of publication over time to identify trends. A secondary goal is to identify the forums in which research has been published. These goals are reflected in the research questions (RQs).

3.3 Search for Primary Studies

The *selection of search terms* is based on [15] and guided by the research questions. The primary studies are typically identified by using search strings on scientific databases or browsing manually through conference proceedings or journals. A common approach to identify the search string

TABLE 1
Research Questions (RQ)

RQ	Motivation
RQ1 (Research Application): Why, in which cloud activities and how have container-based approaches been applied?	While containers can be seen as an alternative to VMs at the infrastructure layer, they are also an application packaging mechanism relevant to platform and software-as-a-service. The mechanisms provided need to be organised in a systematic map of the core architecture concerns identified as follows 1) Motivation: what is the motivation for using containers in the cloud (expected benefits)? 2) Technology Stack: How are container-based systems constructed (application/platform)? 3) Management Services, Cloud Settings and Architecture: How are container-based systems developed and managed? Management services are more platform oriented, whereas cloud settings and architecture capture more abstract, higher-level concerns. 4) Technology Space: What concrete cloud/container technologies are used for construction and management? 5) Application Domain: What are containers in cloud actually used for?
RQ2 (Research Distribution): In which sources and when have studies on container technologies in cloud activities been published?	The topic of this study is broad (covering cloud, software engineering, distributed systems and operating systems) in terms of communities affected and there should be a number of venues to publish the related studies. This information can help us to identify the leading publication venues where the authors can better disseminate their research results and the trend of the number of published studies in this topic.
RQ3 (Maturity): What is the degree of maturity of the field?	The research approaches and evaluation methods tell about the maturity of the field, e.g., whether significant empirical studies have been carried out to establish the value in practice or whether the focus is still on investigating technical problems. The relationship of contribution types, e.g., solution proposals versus experience reports versus reviews as example can answer maturity questions.
RQ4 (Trends): What are the concerns and what is the future research agenda?	The aim is to understand and reveal the research gaps and identify future directions. This is a recent concern and the field is still maturing. Questions therefore arise as to what open questions are and what are the remaining challenges for the future.

is to structure them in PICO terms, which takes into account the research questions. Keywords for the search string can be taken from each aspect of the structure. It is worth noting that, differently from what is suggested by Petersen et al. [15], we do not consider specific outcomes or experimental designs in our study. We avoided this restriction since we wanted a broad overview of the research area as a whole. If we had only considered certain types of studies the overview could have been biased and the map incomplete. Some sub-topics might be over- or under-represented for certain study methods. This difference is also reflected in the search string

$$(\text{cloud}^* \vee \text{PaaS}) \wedge (\text{container}^*) \wedge (\text{orchestrate}^* \vee \text{cluster}^* \vee \text{manage}^*),$$

where '*' matches lexically related terms. Based on the PICO criteria, we chose for 1) Population, here specifically the Technology perspective, the search string *(cloud* OR PaaS) AND (container*) AND (orchestrate* OR (cluster* OR manage*))*. Initially, further PICO categories were considered, but not applied in the search term:

1) Population-Product perspective: Docker OR Kubernetes OR Diego OR Rocket OR LXC OR ...
2) Intervention: experimental OR empirical OR technical
3) Comparison: n/a
4) Outcome: framework OR theory OR architecture OR design OR language OR use case OR case study

The aspects 1 (Product perspective), 2 and 4 were not applied to avoid any incompleteness, but have been considered in the following inclusion/exclusion consideration (part III.D below).

Given that this is a recent concern in cloud computing, the corresponding forums are possibly not fully indexed, causing the need for an initial wider, partly manual search. We started with a wider search (i.e., reduced list of terms as suggested above) to establish an overall body of research, which we then narrowed down towards focus and quality.

The choice of databases we considered is: IEEE Xplore, ACM Digital Library, Science Direct, ISI Web of Science, SpringerLink, INSPEC, EI Compendex, DBLP. Given the recency of the field and concerns with indexing, Google Scholar played the key role for the initial selection before the inclusion and exclusion stage.

Since we used our primary search criteria on title and abstract, this resulted in a high number of irrelevant studies, which were further refined with a secondary search and manual screening-focusing on relevance and resulting then in 46 studies after inclusion/exclusion and quality control.

3.4 Screening of Papers for Inclusion/Exclusion

The *Initial Selection* step includes screening of titles and abstracts of potential studies. We use inclusion and exclusion criteria to exclude studies that are not relevant to answer the research questions. The criteria below show that the research questions influence the inclusion and exclusion criteria:

TABLE 2
Publications Selected-Reference Data

S1. Elastic Application Container: A Lightweight Approach for Cloud Resource Provisioning. *He, Guo, Guo, Wu, Ghanem, Han*. Adv Information Netw, Appl Conf. 2012
S2. Distributed Cloud Storage Services with FleCS Containers. *Yoon, Ravichandran, Gavrilovska, Schwan*. Open Cirrus Summit. 2011
S3. Virtualization versus Containerisation to support PaaS. *Dua, Raja, Kakadia*. Intl Conf on Cloud Engineering. 2014
S4. Skyport Container-based execution environment management for multi-cloud scientific workflows. *Gerlach, Tang, Keegan, Harrison, Wilke, Bischof, Meyer*. Intl Workshop on Data-Intensive Computing in the Cloud. 2014
S5. Containers and Cloud: From LXC to Docker to Kubernetes. *Bernstein*. IEEE Cloud Computing. 2015
S6. Containerisation and the PaaS Cloud. *Pahl*. IEEE Cloud Computing. 2015
S7. Virtualization Driven Mashup Container in Cloud Computing PaaS Model. *Bheda, Thaker*. Intl Conference on Computer Communication and Networks. 2011
S8. vApp: A Standards-based Container for Cloud Providers. *Schmidt, Grarup*. ACM SIGOPS Operating Systems Review. 2010
S9. Containers and Clusters for Edge Cloud Architectures a Technology Review. *Pahl, Lee*. Intl Conference on Future Internet of Things and Cloud. 2015
S10. FlexTuner: A Flexible Container-based Tuning System for Cloud Applications. *Yu, Zou, Tang, Liu, Teng*. Intl Conf on Cloud Engineering. 2015
S11. MultiBox: Lightweight Containers for Multi-Cloud Deployments. *Hadley, Elkhatib, Blair, Roedig*. EGC Workshop. 2015
S12. Self-Adaptive Containers: Interoperability Extensions and Cloud Integration.*Huang, Knottenbelt*. UIC-ATC-SCALCOM '14 Associated Workshops. 2014
S13. Flexible Network Address Mapping for Container-based Clouds. *Kim, Lee, Ben-Ami, Nam, Janak, Schulzrinne*. Conf on Network Softwarization. 2015
S14. Cross-Platform and Cloud-Based Access to Multiple Particle Accelerator Codes Via Application Containers. *Bruhwiler, Nagler, Webb, Andonian, Harrison, Seung, Moeller*. Particle Accelerator Conf. 2015
S15. About Automatic Benchmarking of IaaS Cloud Service Providers for a World of Container Clusters. *Kratzke, Quint*. Journal of Cloud Computing Research. 2015
S16. A Container-based Elastic Cloud Architecture for Real-Time Full-Motion Video (FMV) Target Tracking. *Wu, Chen, Blasch, Liu, Chen, Shen*. IEEE Applied Imagery Pattern Recognition Workshop. 2014
S17. Container-based orchestration in cloud: state of the art and challenges. *Tosatto, Ruiu, Attanasio*. Intl Conf on Complex, Intelligent, Software Intensive Systems. 2015
S18. A REST Service Framework for Fine-Grained Resource Management in Container-Based Cloud. *Li, Tang, Chou*. IEEE Intl Conference on Cloud Computing. 2015
S19. A Performance Isolation Analysis of Disk-intensive Workloads on Container-based Clouds. *Xavier, De Oliveira, Rossi, Dos Passos, Matteussi, De Rose*. Intl Conf on Parallel, Distributed, and Network-Based Processing. 2015
S20. The SPD approach to deploy service-based applications in the cloud. *Yangui, Tata*. Concurrency and Computation: Practice and Experience. 2014
S21. Large-scale cluster management at Google with Borg. *Verma, PedrosL, Korupolu, Oppenheimer, Tune, Wilkes*. European Conference on Computer Systems. 2015
S22. Harbormaster: Policy Enforcement for Containers. *Zhang, Marino, Efstathopoulos*. Intl Conference on Cloud Computing Technology and Science. 2015
S23. Integrating Containers into Workflows: A Case Study Using Makeflow, Work Queue, and Docker. *Zheng, Thain*. Intl Workshop Virtualization Techn in Distr Comp. 2015
S24. Performance Evaluation of Microservices Architectures using Containers. *Amaral, Polo, Carrera, Mohomed, Unuvar, Steinder*. Intl Symp Network Comp & Appl. 2015
S25. Umbrella: A Portable Environment Creator for Reproducible Computing on Clusters, Clouds, and Grids. *Meng, Thain*. Workshop Virtualization Techn in Distr Comp. 2015
S26. A Lightweight Virtualization Cluster Reference Architecture Derived from Open Source PaaS Platforms. *Kratzke*. Open J Mob Comput Cloud Comput. 2014
S27. Docker. *Anderson*. IEEE Software. 2015
S28. Concerning Containers' Connections: on Docker Networking. *Kereki*. Linux Journal. 2015
S29. Exploring Containers for Scientific Computing. *Gomes, Pina, Borges, Martins, Dias, Gomes, Manuel*. Iberian Grid Infrastructure Conference. 2014
S30. Hypervisors versus Lightweight Virtualization: a Performance Comparison. *Morabito, Kjallman, Komu*. Intl Conf on Cloud Engineering. 2015
S31. Dynamic Tailoring and Cloud-based Deployment of Containerized Service Middleware. *Saez, Andrikopoulos, Sanchez, Leymann, Wettinger*. Intl Conf on Cloud Comp. 2015
S32. Performance evaluation of containers for HPC. *Ruiz, Jeanvoine, Nussbaum*. European Conference on Parallel Processing. 2015
S33. The Role of Container Technology in Reproducible Computer Systems Research. *Jimenez, Maltzahn, Moody, Mohror, Lofstead, Arpaci-Dusseau, Arpaci-Dusseau*. Intl Conf on Cloud Engineering. 2015
S34. A DevOps Approach to Integration of Software Components in an EU Research Project. *Stillwell, Coutinho*. Intl Workshop on Quality-Aware DevOps. 2015
S35. Container-based Operating System Virtualization: A Scalable, High-performance Alternative to Hypervisors. *Soltesz, Ptzl, Fiuczynski, Bavier, Peterson*. OS Review. 2007
S36. Performance Evaluation of Container-based Virtualization for High Performance Computing Environments. *Xavier, Neves, Rossi, Ferreto, Lange, De Rose*. Intl Conf on Parallel, Distributed, and Network-Based Processing. 2013
S37. A Discrete-Time Feedback Controller for Containerized Cloud Applications. *Baresi, Guinea, Leva, Quattrocchi*. Symposium on Foundations of Software Engineering. 2016
S38. MicroCloud: A Container-based Solution for Efficient Resource Management in the Cloud. *Baresi, Guinea, Quattrocchi*. Smart Cloud Conf. 2016
S39. Toward a Standard Interface for Cloud Providers. *Fu, Liu, Chu, Hu*. IEEE Internet Computing. 2016
S40. Container and Microservice Driven Design for Cloud Infrastructure DevOps. *Kang, Le, Tao*. Cloud Engineering Conf (IC2E). 2016
S41. Flexible Container-Based Computing Platform on Cloud for Scientific Workflows. *Liu, Aida, Yokoyama*. Cloud Computing Conf. 2016
S42. Auto-tuning Performance of MPI Parallel Programs Using Resource Management in Container-based Virtual Cloud. *Ma, Wang, Tak, Wang, Tang*. Cloud Comp Conf. 2016
S43. Design of an IoT Cloud System for Container Virtualization on Smart Objects. *Puliafito, Villari*. Advances in Service-Oriented and Cloud. 2016
S44. A Container-based Edge Cloud PaaS Architecture based on Raspberry Pi Clusters. *Pahl, Helmer, Miori, Sanin, Lee*. Symp FI Things and Cloud. 2016
S45. Container-Based Cloud Virtual Machine Benchmarking. *Varghese, Subba, Thai*. Cloud Engineering Conf (IC2E). 2016
S46. Container based Video Surveillance Cloud Service with Fine-Grained Resource Provisioning. *Zhang, Ma, Fu, Yang, Jiang*. Cloud Computing Conf. 2016

Inclusion and Exclusion Criteria	
Inclusion	The abstract explicitly mentions containers-in general or in the context of cloud computing. From the abstract, we can deduce that the focus of the paper contributes to chosen research focus. Abstract/keywords include key terms and from the abstract it is clear that a contribution towards containers and their management is made.
Exclusion	The paper lies outside the container context. Container and their management are not part of the contributions of the paper, the terms are only mentioned in the general introductory sentences of the abstract. Literature not peer-reviewed and only in the form of abstract, blog, presentation are excluded.

We started the selection with publications from 2007 with the emergence of the LXC Linux Container technology as a solution for cloud virtualisation and included all relevant publications until the end of 2016.

Category	Description of Research Contribution
Validation Research	Techniques investigated are novel and have not yet been implemented in practice. Techniques used are for example experiments, i.e., work done in the lab.
Evaluation Research	Techniques are implemented in practice and an evaluation of the technique is conducted. It is shown how the technique is implemented (solution implementation) and what are the benefits and drawbacks of the implementation (implementation evaluation).
Solution Proposal	A solution for a problem is proposed, the solution can be either novel or a significant extension of an existing technique. The potential benefits and the applicability of the solution is shown by a small example or a good line of argumentation.
Philosophical Papers	These papers sketch a new way of looking at existing things by structuring the field in form of a taxonomy or conceptual framework.
Opinion Papers	These papers express the personal opinion of somebody whether a certain technique is good or bad, or how things should be done. They do not rely on related work and research methodologies.
Experience Papers	Experience papers explain on what and how something has been done in practice. It has to be the personal experience of the author.

The *Final Selection* step is a validation scan of the studies, considering methods for cloud container orchestration and tool support and details of the evaluation approach. At the end, 46 studies were selected, listed in Table 2.

Qualitative Assessment of Included Studies. For the 46 included studies, we primarily focused on the technical rigor of content presented. We based our assessment on having peer-reviewed contributions and selecting those where the content actually matches the title and abstract based initial selection.

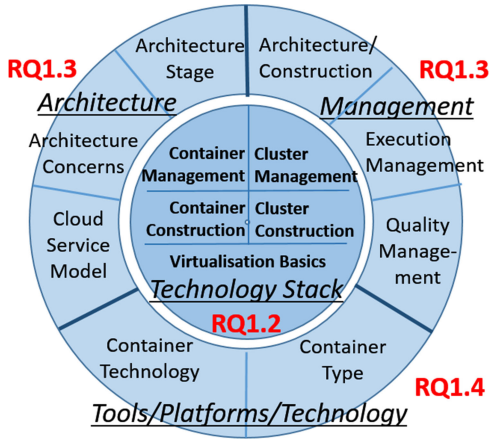

Fig. 3. Categories of cloud container orchestration concerns.

Data Extraction and Synthesis. In order to record extracted data from the selected studies, we followed [15] using a structured format based on characterisation dimensions that cover technical, domain-specific categories as well as generic study classification categories.

3.5 Keywording of Abstracts (Classification Scheme)

Key to our process is keywording to develop the classification scheme and ensuring that the scheme takes the existing studies into account. First, the reviewers involved read abstracts of papers and looked for keywords and concepts that demonstrate the contribution and also the context of each paper. The set of keywords from the different papers considered were then combined into a classification framework to develop a high level understanding of the subject, which is representative of the underlying population.

We combined a number of top-level categories, both technical domain-specific concerns as well as those on the organisation of the research. The generic categories are adopted from the literature, e.g., the contribution type, as in the table below.

The remaining categories are specific to container and cloud technology and aim to answer the research questions (specifically RQ1.1 to RQ1.5) in a targeted way.

- Generic Concerns: Research Contribution, Evaluation Method, Forum, Community, Domain, and Motivation for Technology Adoption.
- Technology-Specific Concerns: Technology Stack, Management Services, Architecture Setting, and Tools/Platforms/Technology.

This technical perspective is highlighted in Fig. 3 where these technical concerns are visualised. The four main concerns *Technology Stack, Architecture, Management* and *Tools/Platforms/Technology* are aligned with the research questions RQ1.2 to RQ1.4, whereby we split RQ1.3 into a conceptual, methodology-oriented *Architecture* aspect and a platform-oriented *Management* aspect that covers the available support services. The concerns are centred around the *Technology Stack* which is about the container technology itself and its internal mechanisms, whereas the surrounding three others are about the cloud environment in which containers are used and supported. The categorisations within the top-level concerns have emerged from the keyword

TABLE 3
Classification Framework

Contribution Type	Solution proposal (Architecture, Framework, Methodology, Library), Evaluation research, Validation research, Experience report, Review (SOTA review, Technology, SLR)			
Evaluation Method	Case study, Mathematical proof, Experience report, Example application, Controlled experiment			
Forum	Journal, Magazine, Conference/Workshop			
Technology Stack	Virtualisation Basics	virtualisation/VM, isolation, hypervisor, OS, control group, namespace		
	Container Construction	Activity construction, application packaging/assembly, provisioning, image building; Container type mashup, micro, meta		
	Container Management	management, communication, application execution		
	Cluster Construction	definition, construction		
	Cluster Management	management, communication, execution		
Management Services	Architecture/ Construction	selection, integration, arch construction/composition, topology, microservice architecture, interoperability/heterogeneity, migration/cloudification, fault handling, adaptivity		
	Execution Management	orchestration, configuration, installation, provisioning, start-up, delivery, load management, scheduling, network address mapping, network management (routing, proxy)		
	Quality/SLA Management	Concern/ Type:	monitorable	performance, elasticity, security, workload, size/volume, resource utilisation, compliance, startup time, reliability, memory use
			non-monitorable:	resilience, portability, interoperability
			testable:	integration, scalability, configurability
		Activity	tuning, self-benchmarking, self-adaptivity	
Architecture Setting	Deployment Stage	requirement, design, deployment, migration, DevOps		
	Architecture Concern	flexibility, modularity, (self-)adaptiveness, integration, interoperability, quality		
	Cloud Setting	single/private, multi/cross/hybrid, edge/fog, hosted, loT-cloud, clustered; laaS, PaaS, SaaS, XaaS		
Tools/ Platforms / Technology	Technology	Docker, Kubernetes, Diego, Rocket, CoreOS, LXC, OpenVZ, STXXL, MCSTL, Intel TBB, Xen, OVF, EAC		
	Container Type	Container, Cluster manager, Unikernel		
Domain	Computing	big data, video, workflow management, HPC, Web application, disk-intensive workloads, resource virtualisation		
	Non Computing	health, physics/science, media, education		
Community	Distributed Systems, Cloud and Big Data, Software Engineering, Autonomic Computing, Network/OS, Application			

extraction and have been aligned with the 4 top-level RQ1-oriented categories.

A first-round keyword extraction has been used to validate the categories, i.e., that concrete terms extracted from the studies occur as instances of the categorisation scheme.

3.6 Data Extraction and Mapping of Studies (Systematic Map)

When the classification scheme is defined, the actual data extraction is the process of categorising the relevant studies into the scheme[1] The classification scheme has evolved during the data extraction (adding new categories and merging or splitting existing ones), based on further input processing and feedback received from independent experts. Data during this process has been gathered in a spreadsheet table to document the data extraction process. From the final table, the frequencies of publications in each category can be calculated. This allows to see which categories have been emphasized in past research and thus to identify gaps and possibilities for future research. We combined two maps-on generic research concerns and on domain-specific technology concerns. For the two maps, we use different ways of presenting and analysing the results. Common generic map targets are: Trends (by year), Forums, and Frequency (by topic). The technology map targets construction aspects (Technology Stack) and management aspects (Architecture and Management Services or Tools, Platforms and Technologies).

The mapping is here from an enriched classification framework to the current research coverage (as shown by the selected papers). The map is illustrated in Section 5 using statistics in form of tables and pie charts, showing the frequencies of publications in each category.

1. Our actual extracted data is provided at http://www.inf.unibz.it/~cpahl/CCT/Cloud_Container_Technology-a_State-of-the-Art_Review.htm.

4 A Classification Framework for Cloud Container Technologies

We first introduce a reference model for an architecture-centric classification of cloud container technologies that helps to demonstrate current research at a conceptual level and identify trends and research directions.

We propose this classification framework, see Table 3, to categorise the primary studies. This framework uses common terms from software engineering methods with methodological support, architecture, tool support and applications.

- The top-level headings (column 1) were already discussed in their alignment with the research questions and illustrated in Fig. 3.
- The subheadings (columns 2 to 4, if applicable) were identified, taking into account the research questions, after a first-round scan of selected studies in order to ensure the validity of the framework. These align with categorisations found in technology reviews [13] for the technology stack, management services, cloud architecture settings and the cloud technologies.
- The concrete terms (last column on the right-hand side of each row) are the terms extracted from the study. We organised them manually into the subheadings above.

The framework has initially undergone several iterations between the authors of this study and have then been validated by five external experts at three organisations in three countries. This has resulted in some corrections and amendments of the first version based on the extraction.

5 Results and Visualisation

Table 4 overviews all selected studies based on the most important categories. The table positions each study using the classification framework, but also to obtain a first overview of the coverage of all studies together. For the aspects 'technology stack' and 'management', we only capture the focus as in these categories multiple occurrences of terms were possible (for which there is not sufficient space in the table). For the architecture setting and the research aspects, we listed all terms extracted. It is worth noting that review papers (a) tend to cover the technology stack more widely and solution papers tend to focus more on specific layers, (b) there is more work on deployment and management services than design and architecture concerns, (c) there is an equal spread between IaaS and PaaS focus, and (d) that mainly experiments and case studies are used for evaluation.

We categorise the selected papers in terms of concerns such as publication format, forum and technical contribution. We also present the key terms extracted from the studies. The results are discussed and the validity of the results and their impact for future research is addressed. The two maps-generic and technology-specific-are discussed separately in this section. We hereafter present the results of our study in more detail (generic and domain-specific maps are presented separately). We also discuss the validity of our results and their potential impact for future research.

5.1 Overview of Primary Studies-Generic Attributes

In order to examine the current state of research on cloud container orchestration, the following questions apply:

- When did research on container technologies become active in the cloud computing community?
- What are the fora in which work on cloud container technologies has been published? On which cloud-related communities does the focus lie?
- How is cloud container technologies research reported and what is the maturity level of the research in this field within cloud computing?

5.1.1 Temporal Overview of Studies

With only a few studies on container orchestration in the early years after the LXC introduction, there has been a dramatic increase since the emergence of tools like Docker, signalling a significant concern, see Fig. 4. The 2007 paper '*Container-based operating system virtualisation: a scalable, high-performance alternative to hypervisors*' is however the most cited one. It can be considered a pioneer in looking at containers as a solution for cloud computing, then confirmed by the interests raised in that paper three years later. This paper brings container technology into the cloud research domain. However, there was not much open source technology in existence that researchers could consider. As a consequence, only after the recent introduction of Docker containers, researcher picked up this direction in research more strongly.

5.1.2 Publication Fora/Communities and Formats

We distinguish communities and formats.

Fora/Communities. We have categorised the publication fora into the computing fields as follows (Fig. 5): *Software Engineering, Autonomic Computing, Distributed Systems, Cloud and Big Data, Network/OS, Application.*

Publications did occur mainly in the now established cloud computing field, as containers can play an important role in cloud (especially PaaS) virtualisation. Other areas include the networks, distributed systems and software engineering communities. We can note that there is not much work in terms of software development for containers and potentials of auto-scaling with containers.

Formats. Regarding the sources, we recognise and distinguish the following peer-reviewed publication formats: journals, magazines, conferences and workshops, and books, see Fig. 6. At this formative stage, we can see mainly workshops and conferences, with magazines and journals often being survey papers. Researchers go mainly for conferences/workshops, as they can provide feedback and publish results in a shortest time, which is of high value in this fast developing area.

5.1.3 Research Methods

Contribution Type. In Fig. 7, the primary studies are summarised according to their contribution type. Solution proposals dominate, with little on validation and evaluation at this (formative) stage. Reviews are technology reviews rather than SLRs. More comparisons among existing solutions are needed (for which our classification scheme can be a compass to organise such solutions).

Evaluation Method. Given the relative immaturity of the domain, the evaluations lack detailed experience reports and proofs, while sample implementations as experience reports and some controlled experiments have been

TABLE 4
Study Comparison-Selected Categories.

	Technology Stack					Management Services		Quality Manag-ern			Architecture Setting			Research Methodology	
Study No.	Virtualisation Basics	Container Construction	Container Management	Cluster Construction	Cluster Management	Architect/Construct	Execution Management	monitorable	non-monitor	testable	Deployment Stage / Context	Architecture Concern	Cloud setting	Contribution type	Contribution type
1	X	X	X	X	X	X	X	X			deployment	flexibility, modularity	laaS	Solution	Experiment
2	X	X					X				deployment, migration	flexibility, integration		Solution	Experiment
3	X	X	X				X	X			design, deployment	flexibility	PaaS	Review	Experiment
4		X					X				deployment	flexibility, adaptive		Solution	Example
5	X										deployment	interoperability	PaaS	Review	
6	X	X	X	X	X	X	X	X			deployment, migration	flexibility, modularity	PaaS	Review	
7		X				X	X		X	X	deployment	flexibility, adaptive	PaaS	Solution	
8		X				X					deployment	integration	laaS	Solution	Example
9	X	X	X	X	X	X	X				deployment, DevOps	modularity	PaaS	Review	
10			X			X	X	X			deployment	flexibility	laaS	Solution	Case study
11			X			X	X	X			migration	flexibility	laaS	Solution	Case study
12			X			X		X		X	deployment	adaptive	laaS	Solution	Case study
13	X		X				X				deployment	integration	laaS, PaaS	Solution	Experiment
14		X				X					migration, deployment	flexibility	PaaS	Solution	Case study
15	X			X	X	X		X			design	quality	laaS	Solution	Case study
16	X		X				X	X			deployment	flexibility	laaS	Experience	Case study. Experiment
17	X	X	X				X	X	X	X	deployment	flexibility, quality	laaS, SaaS	Review	
18			X		X		X	X	X	X	deployment	integration	PaaS	Solution	Experiment
19	X	X						X		X	deployment	quality	laaS, PaaS	Review	Experiment
20		X	X			X	X	X			requirement, design, deployment	flexibility	PaaS	Solution	Case study
21	X	X		X	X	X	X	X			deployment	flexibility	PaaS, SaaS	Solution	Experiment
22	X		X		X			X			requirement, design, deployment	flexibility, quality	PaaS	Solution	Experiment
23	X		X				X	X			design, deployment	integration	PaaS, SaaS	Solution	Case study. Example, Experiment
24	X		X			X	X	X			design, deployment	flexibility, modular., interop.	PaaS	Evaluation	Experiment
25	X					X	X		X		deployment, migration	interoperability, integration	laaS, PaaS	Solution	Case study. Experiment
26				X	X	X			X		requirements, design	modularity	laaS, PaaS	Solution	
27	X	X	X								design, deployment	flexibility, quality	PaaS	Solution	
28	X		X				X			X	deployment	integration	laaS	Experience	Example
29	X	X	X					X	X		design, deployment	integration	laaS	Solution	
30	X	X	X			X	X	X			deployment	modularity	laaS	Solution	Experiment
31											design, deployment	quality	PaaS	Solution, Evaluation	Experiment
32	X	X	X		X	X		X	X		deployment	integration, quality	laaS	Evaluation	Experiment
33			X					X			deployment	integration, quality	laaS	Solution	Case study
34		X	X				X	X		X	design, deployment, DevOps	integration	laaS	Solution	Case study
35	X	X	X				X	X			deployment	flexibility, integration, quality	laaS	Solution,Review	Experiment
36	X				X			X			deployment	quality	laaS	Evaluation	Experiment
37	X	X	X			X	X	X		X	deployment	adaptive	laaS	Solution	Experiment
38	X	X	X			X	X	X			deployment	adaptive	laaS	Solution	Experiment
39	X	X				X		X	X		deployment	quality	laaS, PaaS	Review	
40	X	X	X			X		X	X	X	DevOps	flexibility, modularity, adaptive	PaaS	Solution. Review	Experiment
41			X		X		X	X			deployment	quality	laaS	Solution, Evaluation	Experiment
42		X	X		X		X	X			deployment	quality	laaS	Solution	Experiment
43	X		X	X	X		X	X			design, deployment	flexibility, integration, interop	PaaS	Solution	Example
44											deployment	flexibility, integration	laaS, PaaS	Solution	Experiment
45	X						X	X			deployment	quality	laaS	Solution, Evaluation	Experiment, Case Study
46			X				X	X			deployment	flexibility, quality	laaS	Solution, Validation	Example, Experiment

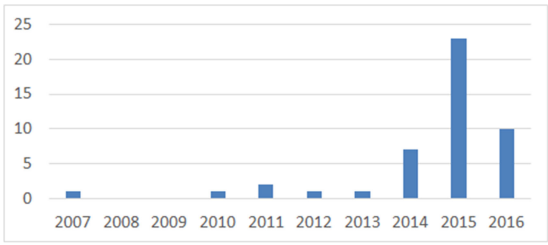

Fig. 4. Study distribution by Year. At the time of writing, the amount of (already indexed) paper published in 2016 is lower than that of 2015. This is most probably because of the delays induced by the online publication of proceedings, by the review processes in journals, and by online indexing leaving the 2016 list of papers inevitably incomplete.

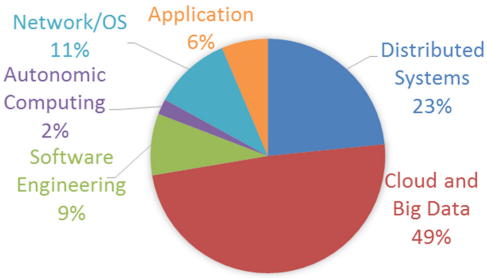

Fig. 5. Study distribution by community.

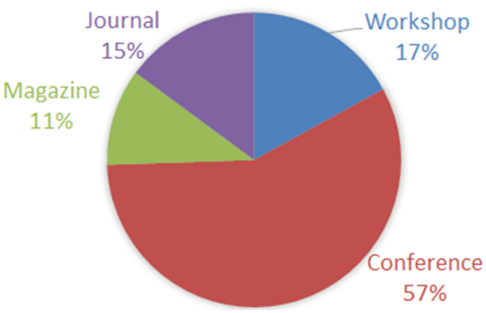

Fig. 6. Study distribution by publication format.

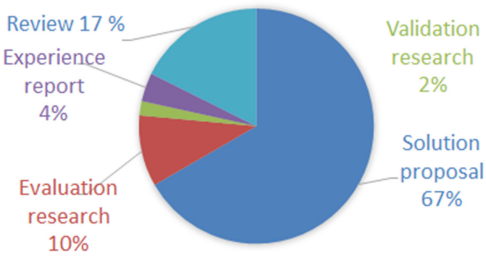

Fig. 7. Study distribution by contribution type.

reported-which matches the solution proposals as the contribution types. Fig. 8 looks at the evaluation methods used. Figs. 10, 11, 12, 13, 14, 15, and 16 complement this by looking more specifically at the technical contribution itself from different perspectives. No mathematical/theoretical foundations are behind the solutions we analysed (as all of them are technological solutions, rather than theoretical approaches). This provides a pointer for further investigation/future work on providing container orchestration with theoretical foundations.

Summary. We can observe a relatively young field, gaining more and more interest over the last years, especially (for conferences) in the cloud domain..

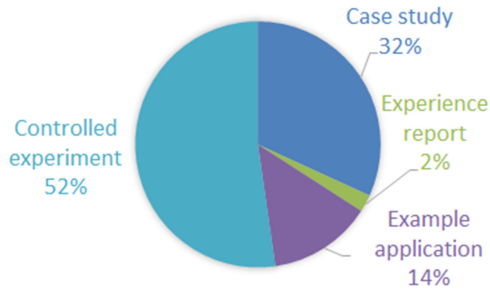

Fig. 8. Study distribution by evaluation method.

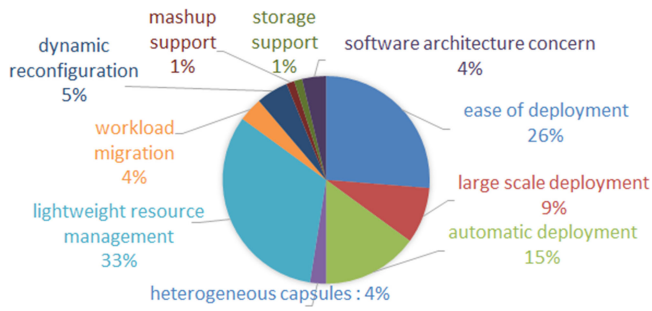

Fig. 9. Study distribution by motivation.

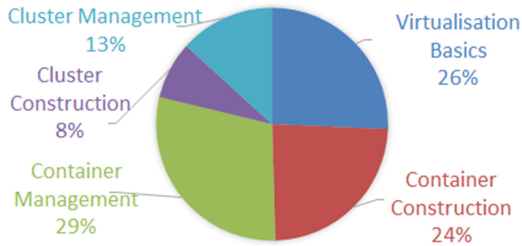

Fig. 10. Study distribution by technology stack layer.

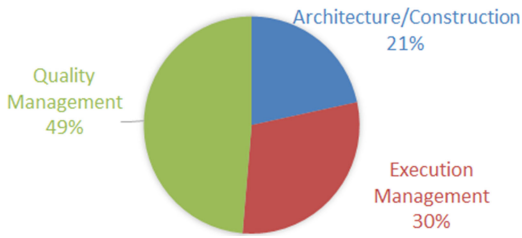

Fig. 11. Study distribution by management service.

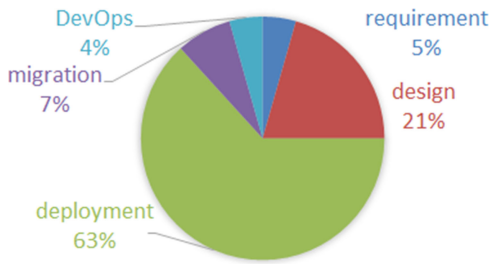

Fig. 12. Study distribution by stage.

5.2 Overview of Primary Studies-Domain-Specific Attributes

Now we focus on the technical aspects of the classification. We will visualise the data again in pie-charts (%-based) and provide the respective total numbers in tables. We also

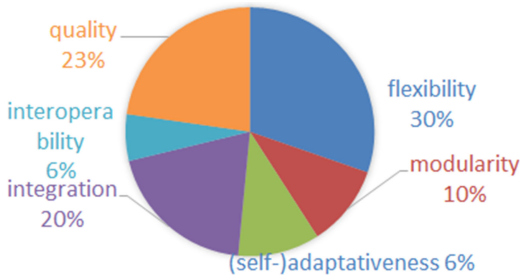

Fig. 13. Study distribution by architecture concerns.

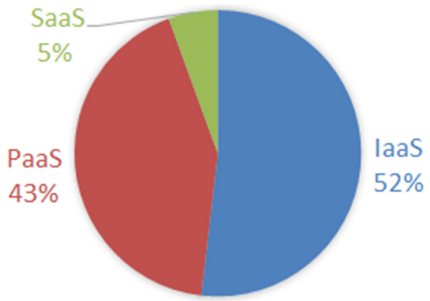

Fig. 14. Study distribution by cloud delivery model.

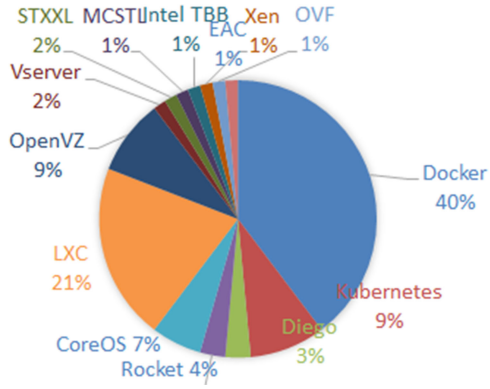

Fig. 15. Study distribution by container technology.

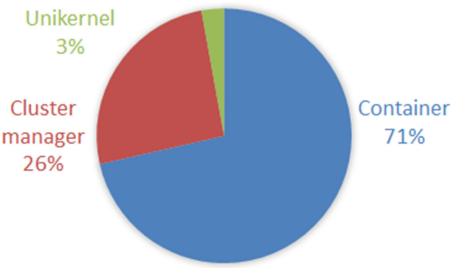

Fig. 16. Study distribution by container type.

provide some additional groupings of terms to clarify further the priorities set by research.[2]

5.2.1 Motivation

Ease of deployment and lightweight resource management are seen as the key benefits. Large-scale and automatic deployment are also key motivations, see Fig. 9. Hence, ease of deployment/management of large-scale heterogenous applications are the main drivers (even before lightweight resource management).

5.2.2 Technology Stack

The technology stack covers the layered construction of virtual resources, which links platform to middleware and orchestration aspects within PaaS, Fig. 10. The focus is on container construction and management (more management than construction), with a good amount of work on virtualisation basics, but also some recent work on clusters.

Technology Stack-Virtualisation Basics: The results here are quite obvious (containerisation is a form of virtualisation), but there is some notable interest in isolation. This is mainly geared towards addressing the security concerns that are due to the container-based virtualisation (where containers are essentially guest processes on a shared operating system, and malicious users could cause more security issues with respect to virtual machines managed by hypervisors). However, according to [S3], containers promise the same level of isolation and security as VMs. No dependence on hardware emulation provides performance benefits over full virtualisation, but restricts the number of supported operating systems. As this is not often required by PaaS providers, containers are suited for providing low-overhead isolation. Container features like cgroups can limit device access for containers from a security perspective, but filesystem, network and memory isolation need more attention.

Technology Stack-Container Construction is more about (functional) composition. The focus is on the actual image-based construction (assembly) process.

Technology Stack-Container Management is on the other hand operations and management oriented. Core execution and wider (quality) management are equally considered; there is some interest in communications of multi-container applications.

Technology Stack-Cluster Construction and Management as distributed architectures reflect the previous two observations for containers.

There is a considerable interest in the core virtualisation aspects of containers, looking into the optimisation of performance for storage [S19] and network aspects [S13]. Other directions beyond orchestration are security concerns [11] such as isolation.

Virtualisation Basics	Count
virtualisation/VM	20
isolation	14
OS	5
control group	5
namespace	5
hypervisor	4

Container Construction	Count	Container Management	Count
construction	14	management	22
provisioning	13	application execution	18
application packaging	10	communication	7
image building	5		

2. In the tables, we summarised for each subcategory a ranked list of term occurrences, highlighting the more relevant concepts, such as an interest in isolation properties. Note, that the terms in the categories can occur multiple times across different studies (with at most one occurrence counted per study).

5.2.3 Management Services

The Management Services address both development and operations/management perspectives by looking at the support services that are provided by a container platform. For example, software engineering activities can be supported by middleware/platform services, software architecture and quality assurance, performance engineering and distributed systems concerns. These are categorised into architecture/construction, execution management and quality management, see Fig. 11.

Cluster Construction	Count	Cluster Management	Count
construction	8	management	12
definition	2	execution	5
		communication	4

Architecture/Construction	Count
construction/composition	10
interoperability/heterogeneity	7
topology (cluster/distribution)	4
adaptivity	4
microservice architecture	4
integration	3
selection	2
cloudification/migration	2
fault handling	2
cloud-native	1

Arch/Contr Group	Terms	Count
Selection & Construction	selection + construction/ composition + microservices	17
Integration	integration + interoperability/ heterogeneity	9
Quality Management	adaptation + fault handling	6
Distribution Management	topology	4
Evolution	cloud / migration	2

Management Services-Architecture/Construction: The main focus is on classical single-system architecture concerns, but also on system integration (architecture and distribution/topology management) and some quality and change management can be noted:

A grouping shows that construction of application containers and their integration are the top concerns:

Execution Management	Count
orchestration	12
provisioning	10
start-up	7
load management	7
configuration	4
installation	4
scheduling	4
network address mapping	4
delivery	3
network management (routing, proxy)	3

Integration in environments with interoperability challenges due to heterogeneity is the second strongest group. The benefit of container technology to aid the orchestration of applications in distributed topologies is highlighted in [S9], the emerging trend towards edge cloud computing and its orchestration needs are also noted. Cluster management is instead covered on [S21], [S25] and [S26].

Management Services-Execution Management: the following concerns have been recorded:

This can again be grouped. There is a balance between set up/preparation, core execution management, wider (quality) management and, almost as much, enabling network/infrastructure technology.

While performance and scalability are accepted IaaS concerns, managing performance in container-based application architectures is also a PaaS concern [S24,S32]. Performance is central in all aspects here, from load management to configuration to provisioning and efficient network management.

Management Services-Quality Management results are below: The three categories monitorable/non-monitorable/-testable are split 56/8/6, respectively. Monitorable and testable quality aspects are in the majority and this can be attributed to the large number of experimental studies in the systems community.

Exec Mgmt Group	Terms	Count
Advanced Management	load management + scheduling + orchestration	23
Preparation	configuration + start-up + installation	15
Core Execution	delivery + provisioning	13
Network/ Infrastructure	network management + address mapping	7

There is a balance between infrastructure quality parameters, (external/given/provided) service-level objectives (SLO) and some parameters describing the adaptation mechanism in-between (e.g., elastic), as the grouping below shows:

The fact that the aim is gain in performance (less overhead) through containers as virtualisation mechanisms is noteworthy [S13,S19,S30,S35,S36,S39]. [S1] reports that containers are more resource efficient and more scalable due to the significantly lower RAM consumption (29.4 times smaller than VM). High-performance computing (HPC), where both isolation and performance is needed, benefits since container-based virtualisation provides less overhead than hypervisor-based environments and isolation concerns are less prevalent in HPC due to limited resource sharing needs [S36,S42].

Quality Category	Terms	Count
monitorable	performance	23
	resource utilisation	13
	startup time	10
	elasticity	10
	security	7
	reliability	4
	memory use	4
	workload	3
	size/volume	3
	compliance	1
non-monitorable	portability	5
	interoperability	5
	resilience	2
testable	scalability	6
	configurability	2
	integration	1

Quality Group	Terms	Count
SLA Parameter	performance, compliance, security, reliability	36
Infrastructure Parameter	workload, resource utilisation, startup, memory use, size/volume	33
System/Mgmt Parameter	elasticty	10

Group of Stages/Activities	Count
requirements + design	17
deployment	43
migration + DevOps	8

Architecture Group	Terms	Count
Construction	modularity + integration + interoperability	24
Change	flexibility + self-adaptation	27
Other qualities	[remaining qualities]	15

There is interest both in both SLAs and infrastructures. Concerning SLAs the predominant parameter is performance, while for infrastructures the predominant parameters are resource utilisation and portability. What emerges here is a stakeholder dimension: SLA concerns are important for the consumers/users of the application, infrastructure concerns are related to the providers and the system/management category addresses an internal perspective.

5.2.4 Architecture Setting

Wider software engineering concerns are also covered by our classification scheme, which extends the Architecture/Construction and Management Services, but at a higher level.

- 'Stage' refers to the development stage, Fig. 12, which can be summarised into three groups of concerns:

  The concern is mainly deployment, with less design stage activities and only a few cross-cutting concerns (such as DevOps). This confirms other observations above.

  The operational concerns of deployment and runtime monitoring and development aspects like testing would benefit from a tighter integration. For instance, [S34] shows that the combination of automated testing and deployment improves the speed and efficiency. Continuous deployment and automation of activities in a DevOps-style can be the solution.
- *Architecture Concerns*, see Fig. 13: On a term-by-term basis, 'flexibility' emerges as a key benefit for clouds and containers, but also that applications have to be integrated and run under quality requirements. Fig. 13 is detailed in the table below. We can observe an equal balance between construction and management (change and adaptivity) towards DevOps and continuous development [S20,S40]. Other qualities are less relevant.

5.2.5 Cloud and Container Technology Space

- *Cloud Delivery Model*: we observe a mix of IaaS and PaaS. The data we collected clearly shows that containerisation is useful at the level of IaaS and PaaS, with both IaaS and PaaS almost equally distributed. A few more contributions target IaaS from a virtualisation and orchestration perspective, rather than PaaS with its application packaging focus. IaaS is often present in the analysed papers. Many of them also address virtual machines and compare how containers perform due to having a lighter virtualisation approach [2], [3]. VMs are the traditional model for IaaS, but containers are similar as image-based machines, but also oriented towards PaaS through application packaging aspects, typically running on top of a virtual machine rather than running directly on a host OS [S3].
- *Container Technologies*: Docker and to a smaller extent LXC dominate, other trending ones are covered (Kubernetes, CoreOS, OpenVZ, Diego, Rocket). LXC can be considered the de-facto standard for containerisation on Linux/Unix, while Docker has gained momentum and it is becoming the reference technology for containerisation.

  Open-sourcing, a rich ecosystem with image repositories and community support are here the main drivers of success. The popularity of Docker is probably linked to the open source approach and early release of technology. Though, as Bernstein points out [S5], a more neutral governance/collaboration structure around Docker (a start-up company) and Kubernetes (still controlled by Google) will allow an agreement on a wider common packaging and deployment approach. An industry perspective is added in [S8] by VMware, demonstrating an OVF-based standardised containerisation approach. Standard containers that cleanly separate application providers from infrastructure providers are a key success factor [6].
- *Container Type*: Here, we see largely containers, then clusters (confirming above observations). However, looking at publication times, we could image that containers are more popular than clusters because the latter has been only started very recently with publications almost exclusively from mid 2015, e.g., [S9,S15,S43,S44].

Summary. We can note that container-based solutions target both IaaS and PaaS, with a consolidated interest for single-container solutions, and a growing interest for solutions for clusters of containers. Managing qualities such as performance and resource utilisation are key concerns.

5.3 Completing the Conceptual Map

Furthermore, we looked at the Technology Stack and Management Services categories in more detail, as these were only summarised in the initial study summary in Table 4, where we compared the studies based on some core classification categories. In Figs. 17 and 18, we mapped the generic contribution types (Solution, Evaluation, Experience Report, Review) to the two above specific technical categories

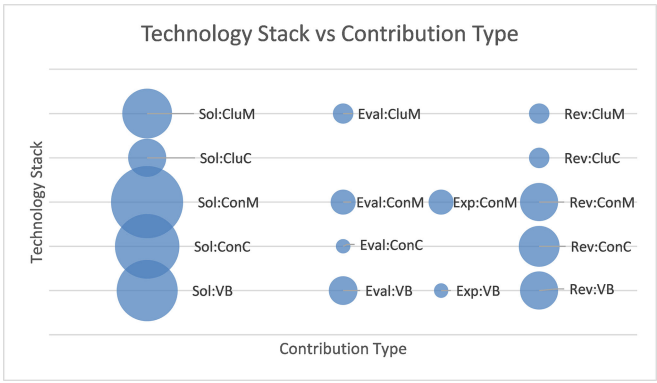

Fig. 17. Contribution type by stack concern. [Sol:Solution, Val:Validation, Eval:Evaluation, Exp: Experience Report, Rev: Review], [VB: Virtualisation Basics, ConC: Container Construction, ConM: Container Management, CluC: Cluster Construction, CluM: Cluster Management].

relevant in the container orchestration context. The bubble size indicates the number of studies of that type.

The Technology Stack view, Fig. 17, shows how containers and container-based clusters are internally constructed:

- mainly solutions are reported (largest bubbles), which primarily cover container construction and management, but also address all other aspects to some extent;
- evaluations focus on construction, less on management;
- equally, experimentation is construction-focussed;
- reviews cover the technology stack more completely.

In the Management Services view, Fig. 18, similar observations can be made:

- mainly solutions are reported, which here more clearly focus on container execution and quality management (regarding the latter, monitorable aspects prevail);.
- evaluation, experimentation and examples are patchy, focussing on the quality management;
- reviews cover all service types more comprehensively.

Summary. The conceptual map links the studies with the classification scheme. For each term, we recorded the number of occurrences of the extracted terms from the studies. Higher numbers of occurrences in each of the categories indicate a stronger research concern and/or stronger consensus

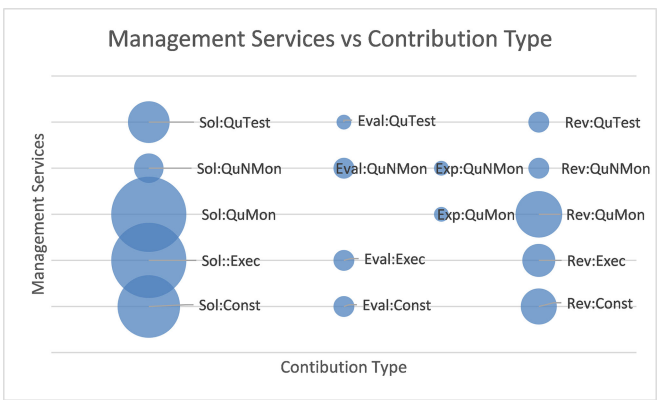

Fig. 18. Contribution type by management service. [Sol:Solution, Val: Validation, Eval:Evaluation, Exp: Experience Report, Rev: Review], [Const: Construction/Architecture, Exec: Execution, QuMon: Quality-Monitorable, QuNMon: Quality-Non-Monitorable, QuTest: Quality-Testable].

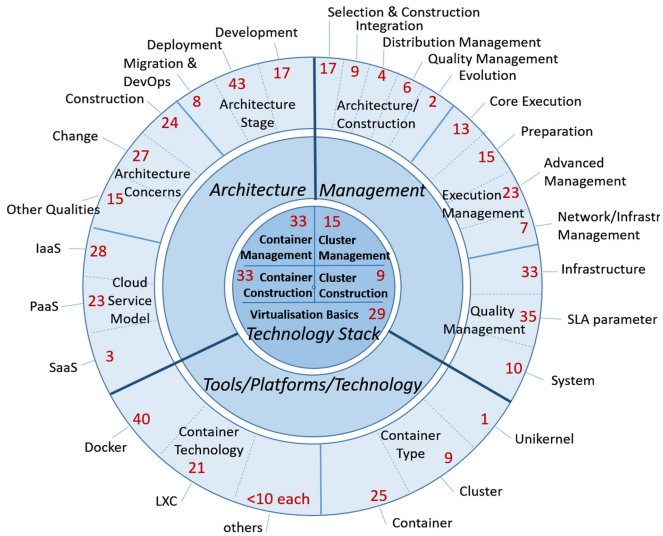

Fig. 19. Key terms extracted, organised into different concerns and presented with the numbers of times these terms occur.

on the importance of the aspect. In Fig. 19, the term occurrences are associated to the technical concerns of the classification scheme. The figure summarises earlier discussions and defines the cloud container technologies space into

- internal construction of containers and their clusters,
- methodological support for construction and architecture,
- infrastructure services for development, execution management and quality control,
- technology and tool platforms.

The relative weight of the individual concerns within each of the four core categories is given. This shows the current focus on developing solutions for runtime management of execution and quality, but also the need to:

- (methodological aspect) evaluate and experiment more,
- (technical aspect) integrate the runtime focus more with development aspects into a coherent framework,
- (quality aspect) broaden the quality concerns beyond the monitorable ones.

Particularly the Architecture and Management categories define what are the expected functions of a cloud container technologies (with methods and techniques).

5.4 Threats to Validity

We discuss threats to the validity of this work in the different mapping study steps.

Threats to the Relevance of the Selected Studies. Innovation in the area emerges from industrial practice. While industrial white papers or blogs as mechanisms to communicate the innovations are available, these are difficult to systematically retrieve and quality assess through SLRs or SMSs. We have therefore complemented the protocol-based results by reviewing relevant technologies such as Docker, cloud and cloud-native architectures and including exemplary blogs in the discussion.

Threats to the Identification of Primary Studies. In our search, we aimed to retrieve as many studies as possible to

avoid any bias. A challenge was to determine the scope of our study, since container orchestration relates to different computing and IT communities including software engineering, distributed systems, operating systems, information systems and cloud. These communities use different terminologies for the same concepts. To cover all and avoid any bias, we searched for the container orchestration term in different contexts. While this approach decreases bias, it significantly increases search effort. To identify relevant studies and ensure an unbiased selection, a review protocol was developed.

Threats to Selection and Data Extraction Consistency. The formulation of the research questions has helped in selecting studies of relevance, as did the Characterisation Framework. However, we did include peer-reviewed magazine contributions and books here to capture trends and activities.

Threats to Data Synthesis and Results. This reliability threat is mitigated as far as possible by having a unified characterisation scheme and following a standard protocol where several steps were piloted and externally evaluated.

6 Conclusions

Containerisation provides cloud application managment based on lightweight virtualisation. The increasing interest in cloud container technologies shows the importance of their management and orchestration in this context. The scientific contributions we reviewed are a mix of technology reviews, solutions and use case architectures (conceptual and implemented). As a good part of this is still conceptual, it can be seen as a sign of the immaturity of an emerging field. This interpretation is strengthened by the fact that only some use case validations and no large-scale empirical evaluations exist. There is also a noticeable imbalance of contribution formats compared to more mature domains:

- Larger number of technical contributions (solution proposals): the number of journal publications is low (with short communications in magazines published only).
- Higher number of use cases than technology solutions: i.e., an emerging technology to be formatively validated through use cases rather than summative evaluations.

What has been demonstrated is the better resource efficiency of containers compared to VMs, and also the increased flexibility as an application management framework. However, proven technologies are lacking to fully support container architectures for cloud environments. An example pointed out in [S21] is failure management. Finding root causes and dealing with anomalous events requires better resource monitoring and log analysis.

We can conclude that the field is moving towards container middleware (and even container PaaS) with isolation, construction, quality management, orchestration and distribution management as core concerns of a container PaaS middleware, but key features such as failure management are still missing. We see here middleware as the core set of features necessary to host and provide applications, facilitating the application and Web tier. A PaaS is a more comprehensive solution encompassing (virtualised) infrastructure features and data(base) storage. Container orchestration is a key concern in the current cloud PaaS context. For instance, the Cloud Native Computing Foundation CNCF (https://cncf.io/) will be integrating the orchestration layer of the container ecosystem. The CNCF uses Kubernetes container cluster orchestration as its containerisation technology to deal with network storage and cluster management. Industry reports, such as [S21] about Google's Borg cluster management, highlight current limitations-e.g., Borg has no first-class mechanism to manage an entire multi-job service as a single entity. Better cluster orchestration support is the required solution. Management becomes inherently distributed, with features like the scheduler, admission control, vertical and horizontal autoscaling, re-packing, periodic job submission, workflow management and archiving. The aim is to scale up the workload and feature set without sacrificing performance or maintainability.

Organizations are packaging applications in containers and need to orchestrate multiple containers across cloud servers:

- The benefits of container-based orchestration include adjustable cluster sizes for the deployment of containers, easier cluster maintenance and also quicker deployment. This is confirmed by the motivations cited, which prioritise the easy, automated container deployment and management in larger settings allowing for flexible migration and reconfiguration.
- Containerisation positively impacts on both development and deployment aspects such as testing and monitoring of container-based applications, which we found both covered by the studies in terms of a mix of architecture design and continuous, quality-driven management.
- The quality distribution shows a strong interest in optimised resource utilisation (effectively a cost factor) and performance, complemented by portability/interoperability and security. Interesting is also the emergence of two equally important perspectives: quality aspects directly observable by the consumers and the quality aspects relevant to the management of the applications within the platform itself.

A theme that emerges is the support of continuous development through containers, joining both construction as well as operations and management. In the cloud, cloud-native platform services for development and deployment do exist, but require advanced PaaS orchestration support.

The trend towards cluster-based orchestration, combined with the interoperability of successful container technologies, also allows the management of highly distributed topologies of smaller virtualised devices beyond centralised clouds as in the edge/fog cloud domain [S9,S21,S25,S26, S43,S44]. Containers can run on single-board devices and can be deployed on clusters of these devices, making them suitable for edge and IoT computing [14].

This results in a need for research beyond the performance and isolation concerns for container virtualisation, particularly regarding *methodological and tool support*:

- A DevOps approach to manage the continuous development and deployment would benefit-the cross-stage nature is obvious. While most papers look at the

two sides separately, the need to link these in a continuous process and to feed back monitored operational data into development becomes obvious.

- Containers represent a progression from virtual machines towards lightweight application management. Lately, there is an observable continuing trend towards serverless architectures and other mechanisms to manage orchestration and deployment complexity such as unikernel technology towards more lightweightness.
- Similar to Docker kick-starting research on container technology as an active open-source environment, a similar impact may be expected with the emergence of similar technologies in the cluster space such as Kubernetes and Mesos. There is a strong increase in research papers from mid 2015 that evidences this. Edge or fog computing is an architectural setting that needs clustering in which containers can help to address interoperability needs.

Acknowledgments

The work presented here was partly supported by project PRA_2016_64 "Through the fog" funded by the University of Pisa and by project "Edge Cloud Orchestration" funded by the Free University of Bozen-Bolzano.

References

[1] A. Brogi, et al., *SeaClouds: An Open Reference Architecture for Multi-Cloud Governance*. Cham, Switzerland: Springer, 2016, pp. 334–338.
[2] C. Dupont, M. Sheikhalishahi, F. M. Facca, and S. Cretti, "Energy efficient data centres within smart cities: IaaS and PaaS optimizations," in *Smart City 360*. Cham, Switzerland: Springer, 2016, pp. 408–415.
[3] W. Felter, A. Ferreira, R. Rajamony, and J. Rubio, "An updated performance comparison of virtual machines and Linux containers," in *Proc. IEEE Int. Symp. Performance Anal. Syst. Softw.*, 2015, pp. 171–172.
[4] P. Jamshidi, A. Ahmad, and C. Pahl, "Cloud migration research: A systematic review," *IEEE Trans. Cloud Comput.*, vol. 1, no. 2, pp. 142–157, Jul.–Dec. 2013.
[5] P. Jamshidi, M. Ghafari, A. Ahmad, and C. Pahl, "A framework for classifying and comparing architecture-centric software evolution research," in *Proc. 7th Eur. Conf. Softw. Maintenance Reengineering*, 2013, pp. 305–314.
[6] G. Kecskemeti, A. C. Marosi, and A. Kertesz, "The entice approach to decompose monolithic services into microservices," in *Proc. Int. Conf. High Performance Comput. Simul.*, 2016, pp. 591–596.
[7] B. Kitchenham, O. P. Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman, "Systematic literature reviews in software engineering–a systematic literature review," *Inf. Softw. Technol.*, vol. 51, no. 1, pp. 7–15, 2009.
[8] N. Kratzke, "About microservices, containers and their underestimated impact on network performance," in *Proc. 6th Int. Conf. Cloud Comput.*, 2015, pp. 165–169.
[9] J. Lewis and M. Fowler, *Microservices*, 2014. [Online]. Available: http://martinfowler.com/articles/microservices.html
[10] C. Liu, B. T. Loo, and Y. Mao, "Declarative automated cloud resource orchestration," in *Proc. 2nd ACM Symp. Cloud Comput.*, 2011, Art. no. 26.
[11] A. Manu, J. Patel, S. Akhtar, V. Agrawal, and K. Murthy, "Docker container security via heuristics-based multilateral security-conceptual and pragmatic study," in *Proc. Int. Conf. Circuit Power Comput. Techn.*, 2016, pp. 1–14.
[12] P. Mell and T. Grance, "The NIST definition of cloud computing," *Recommendations Nat. Inst. Standards Technol.*, Special Publication 800-145, (2011). [Online]. Available: http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf
[13] C. Pahl, "Containerization and the PaaS Cloud," *IEEE Cloud Comput.*, vol. 2, no. 3, pp. 24–31, May/Jun. 2015.
[14] C. Pahl, S. Helmer, L. Miori, J. Sanin, and B. Lee, "A container-based edge cloud PaaS architecture based on raspberry Pi clusters," in *Proc. 4th IEEE Int. Conf. Future Internet Things Cloud Workshops*, 2016, pp. 117–124.
[15] K. Petersen, R. Feldt, S. Mujtaba, and M. Mattsson, "Systematic mapping studies in software engineering," in *Proc. Int. Conf. Eval. Assessment Softw. Eng.*, 2008, pp. 68–77.

Claus Pahl received the MSc degree from the University of Technology in Braunschweig and the PhD degree in computing from the University of Dortmund. He is an associate professor with Free University of Bozen-Bolzano. Prior to his current employment, he was a principal investigator of the Irish Centre for Cloud Computing and Commerce IC4. His research interests include software engineering in service and cloud computing.

Antonio Brogi received the PhD degree in computer science from the University of Pisa, in 1993. He is a full professor in the Department of Computer Science, University of Pisa, Italy, since 2004. His research interests include service-oriented and cloud-based computing, coordination and adaptation of software elements, formal methods, and design of programming languages. He has published the results of his research in more than 150 papers in international journals and conferences.

Jacopo Soldani received the PhD degree in computer science from the University of Pisa, in 2017. He is a post-doc researcher with the University of Pisa, Italy. He is member of the SOCC Research Group. His research interests include service-oriented, cloud and fog computing, adaptation, coordination, and integration of software elements, and formal methods. He has been involved in research projects on cloud and fog computing both at local and EU level.

Pooyan Jamshidi received the BS and MS degrees in computing from Amirkabir University of Technology, and the PhD degree from Dublin City University. He is a postdoctoral researcher with Carnegie Mellon University. Prior to his current position, he was a research associate in the Imperial College London. His general research interests are in software engineering and his focus lies predominantly in the areas of highly-configurable systems, machine learning, and data-intensive computing.