Microservices: A Systematic Mapping Study

Claus Pahl[1] and Pooyan Jamshidi[2]
[1]*Faculty of Computer Science, Free University of Bozen-Bolzano, Bolzano, Italy*
[2]*Department of Computing, Imperial College London, London, U.K.*

Keywords: Mircoservices, Container, Cloud, Systematic Literature Review, Systematic Mapping Study.

Abstract: Microservices have recently emerged as an architectural style, addressing how to build, manage, and evolve architectures out of small, self-contained units. Particularly in the cloud, the microservices architecture approach seems to be an ideal complementation of container technology at the PaaS level However, there is currently no secondary study to consolidate this research. We aim here to identify, taxonomically classify and systematically compare the existing research body on microservices and their application in the cloud. We have conducted a systematic mapping study of 21 selected studies, published over the last two years until end of 2015 since the emergence of the microservices pattern. We classified and compared the selected studies based on a characterization framework. This results in a discussion of the agreed and emerging concerns within the microservices architectural style, positioning it within a continuous development context, but also moving it closer to cloud and container technology.

1 INTRODUCTION

Recently, microservice architectures have been proposed to break up application architectures into independently deployable services that can be rapidly deployed to any infrastructure resource as required (Newman, 2015; Lewis and Fowler, 2014). Microservices are independently deployable, usually supported by a deployment and orchestration framework, e.g., in the cloud, enabling them to deploy often and independently at arbitrary schedules. Microservice deployment and orchestration across the distributed and stacked nature of the cloud are central architecture concerns. Clouds provide a management tool for their flexible deployment schedules and provisioning orchestration needs, particularly, if these are to be PaaS-provisioned.

Research on microservices has addressed both the architectural principles and support as well as the application of the architectural pattern in the cloud. In the cloud, microservices are linked to containers, which are a lightweight virtualisation mechanism used for application packaging, distribution and orchestration at the PaaS layer (Pahl, 2015), even towards edge clouds with smaller resource environments (Pahl and Lee, 2015).

There has not been a systematic literature review (SLR) or mapping study (SMS) of research on microservices that would allow to assess the maturity in general and identifying trends, research gaps and future directions. Given the growing interest in containers and microservices in cloud, there is a need to explore a research agenda. SLRs and SMS identify, classify and synthesize a comparative overview of state-of-the-research and enable (Petersen et al., 2008; Kitchenham et al., 2009). We opt for a systematic mapping study as it is more suitable in mapping out and structuring new areas of investigation.

We aim here to identify, taxonomically classify and systematically compare the existing research body on microservices and their application in the cloud. We have conducted a systematic mapping study of 21 selected studies, published over the last two years since the emergence of the microservices pattern. We classified and compared the selected studies based on a characterization framework.

The research mapping study resulted in a knowledge base of current research approaches, methods, techniques, best practices and experiences used in microservices architecture, with a particular attention to cloud application. This review reveals that microservices research is still in a formative stage. More experimental and empirical evaluation of the benefits is needed. This study also showed a lack of tool support to automate and facilitate cloud microservice.

The results of this study aim to benefit, firstly, researchers in software engineering and cloud computing, who need an identification of relevant studies. Secondly, practitioners interested in understanding the available methods and techniques with tool

Pahl, C. and Jamshidi, P.
Microservices: A Systematic Mapping Study.
In *Proceedings of the 6th International Conference on Cloud Computing and Services Science (CLOSER 2016) - Volume 1*, pages 137-146
ISBN: 978-989-758-182-3

support as well as their constraints and maturity level in supporting microservice architectures.

The remainder of this paper is structured as follows. Section 2 describes background and related research to position contributions of this work. Section 3 explains our research methodology, research questions and scope. Section 4 provides a reference model for state-of-the-research and a characterization scheme. Section 5 presents the results of the mapping study. Section 6 discusses the findings, implications and trends followed by an analysis of its limitations.

2 BACKGROUND

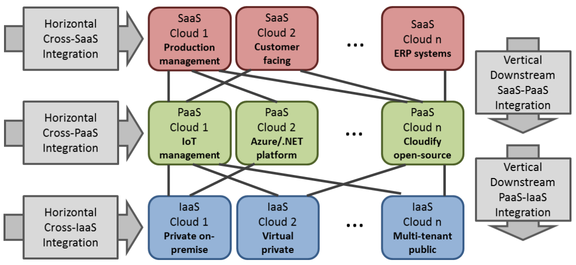
Based on (Newman, 2015; Lewis and Fowler, 2014), microservices are about functional decomposition often in a domain-driven design context. They are characterised by well-defined and explicitly published interfaces. Each service is fully autonomous and full-stack. Consequently, changing a service implementation has no impact to other services as communication takes place using interfaces only. Functional decomposition of an application and the team is the key to building a successful microservices architecture. This achieves loose coupling (REST interfaces) and high cohesion (multiple services can compose with each other to define higher level services or application). Functional decomposition enables for instance agility, flexibility, scalability.

2.1 Lightweight Architectures, Services, Containers and Microservices

Microservices are an architectural pattern emerging out of Service-Oriented Architecture (SOA), emphasising self-management and lightweightness. While this is an architectural principle, container technology has emerged in parallel in cloud computing to provide a lightweight virtualisation mechanism that can serve as an application packaging mechanism for PaaS clouds. Microservice can ideally be packaged, provisioned and orchestrated through cloud.
The cloud can be seen as a distributed and tiered architecture, see Figure 1. The core infrastructure, platform and software application tiers can distributed across multi-cloud environments – based on container-based microservices.

2.2 Need for a Secondary Study

The need for a SLR or SMS entails to identify, classify and compare existing evidence on the use of microservices specifically in cloud environments


Horizontal Cross-SaaS Integration
SaaS Cloud 1 Production management
SaaS Cloud 2 Customer facing
...
SaaS Cloud n ERP systems
Vertical Downstream SaaS-PaaS Integration
Horizontal Cross-PaaS Integration
PaaS Cloud 1 IoT management
PaaS Cloud 2 Azure/.NET platform
...
PaaS Cloud n Cloudify open-source
Vertical Downstream PaaS-IaaS Integration
Horizontal Cross-IaaS Integration
IaaS Cloud 1 Private on-premise
IaaS Cloud 2 Virtual private
...
IaaS Cloud n Multi-tenant public
Figure 1: Cloud Reference Architecture Model.

through a characterization framework. Some technology reviews exist (and also classified as such later), but these concentrated on technology and do not capture research efforts and directions systematically.

3 RESEARCH METHODOLOGY

SMSs reduce bias through rigorous sequence of methodological steps to research literature. They rely on well-defined and evaluated review protocols to extract, analyze and document results. We follow the process presented in (Petersen et al., 2008) 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 characterization framework for the study context.

Table 1: Research Process.

Step	Activity
plan	identify need, specify RQs, define protocol
conduct	select primary studies, extract/synthesize data
document	document observations, analyze threats, report

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

3.1 Planning the Review

Step 1 – Identify the Need for SLR. We have already discussed the need for a SMS in the previous section. We can also make explicit the general goal and scope of the study using the PICO (Population, Intervention, Comparison, Outcome) criteria, see Table 2.
Step 2 – Specifying the Research Questions. As the next activity, we define the research questions to help shaping the review protocol, see Table 3.
Step 3 – Define and Evaluate Review Protocol. We developed a protocol for a mapping study based on (Petersen et al., 2008) and our experience with SLRs

Table 2: PICO criteria.

Concern	Explanation
Population	RQ1: Practical motivation, RQ2: Architecture tasks, RQ3: Methods and techniques, RQ4: Research challenges and future dimensions [all detailed below]
Intervention	Characterization, Internal/external validation; Extracting data and Synthesis
Comparison	A comparison by mapping the primary studies to a characterization framework
Outcome	A characterization framework

Table 3: Research Questions.

Research Question	Motivation
RQ1 What are the main practical motivations behind using microservices?	The aim is to get insight in what are the main reasons for organizations to architect in a microservices style.
RQ2 What are the different types of microservice architectures involved?	The aim is to investigate what are the possibilities for architecting in a microservice style.
RQ3 What are the existing methods, techniques and tool support to enable microservice architecture development and operation?	The aim is to identify and compare existing methods and techniques that support microservice architecture.
RQ4 What are the existing research issues and what should be the future research agenda?	The aim is to understand and reveal the research gaps and identify future directions.

(Jamshidi et al., 2013a; Jamshidi et al., 2013b) that is outlined in this section.

3.2 Conducting the Review

Conducting starts with study selection and resulting in extracted data and synthesized information.

Step 1 – Select Primary Studies (Study Selection and Qualitative Assessment). The search terms used were developed using (Petersen et al., 2008) and guided by research questions. While we initially considered to include terms like ’architecture’ or ’lightweight’, however, ultimately we only used the term ’microservice’ (and variations such as ’micro-service’ as search terms, cf. 4, to avoid excluding any studies in this emerging context. We extracted initially 243 studies from years 2014 to 2015 (incl.) – as of 3 Nov. 2015. The year 2014 was chosen as the term microservice as an architectural pattern is only consistently used since then. Earlier occurrences referred to microservices as somehow small in scale, but without a clear reference to an architectural style or pattern, and were thus excluded.

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 – focusing on title occurrences only and resulting then in 21 studies.

Step 1a – Initial Selection. This step includes screening of titles and abstracts of potential primary studies – performed against inclusion/exclusion criteria

Table 4: Selection Terms.

microservice	micro-service

Table 5: Inclusion/exclusion criteria.

In/Excl	Criteria
Inclusion	• Abstract/keywords include key terms • From the abstract it is clear that a contribution towards microservices and their management is made
Exclusion	• Type: literature only in the form of abstract, blog, presentation are excluded • Papers with microservices terms only in abstract or with different meaning

Step 1b – Final Selection. This is based on a validation scan of the studies, methods for microservices and tool support and details of the evaluation approach. At the end, 21 studies were selected.

Step 1c – Qualitative Assessment of Included Studies. For the 27 included studies, we primarily focused on the technical rigor of content presented. We based our qualitative assessment on factors like General Assessment (G) and Specific Assessment (S). Quality scores provided us with a numerical quantification to rank the selected studies, but given the recency of the development, we also included books and theses/dissertations as long as there is evidence of a review being carried out.

Steps 2 and 3 – Data Extraction and Synthesis. In order to record extracted data from the selected studies, we follow (Petersen et al., 2008) using a structured format based on characterization dimensions.

4 A MICROSERVICE ARCHITECTURE FRAMEWORK

We first introduce a reference model for an architecture-centric classification of microservices that helps to demonstrate current research at a conceptual level and identify trends and research directions. We follow (Zimmermann, 2009) in his definition of an architectural style and present the microservices style as a collection of principles and patterns. We then present a framework to characterize individual

Table 6: Publications selected.

1	Rodrguez Molina, J. (2015). Distribution of microservices for hardware interoperability in the Smart Grid (Dissertation, ETSIS_Telecomunicacion).
2	Scholz, N. J. (2015). Evaluation of Microservices as Extension of Established Component Technologies. (Diss RWTH Aachen).
3	Bak, P., Melamed, R., Moshkovich, D., Nardi, Y., Ship, H., and Yaeli, A. (2015). Location and Context-Based Microservices for Mobile and Internet of Things Workloads. IEEE Intl Conf on Mobile Services (MS), pp. 1-8.
4	Nycander, P. (2015). Learning-Based Testing of Microservices: An Exploratory Case Study Using LBTest.(Dissertation KTH Stockholm)
5	Stubbs, J., Moreira, W., and Dooley, R. (2015). Distributed Systems of Microservices Using Docker and Serfnode. In International Workshop on Science Gateways (IWSG), 2015.
6	Levcovitz, A., Terra, R., and Valente, M. T. (2015). Towards a Technique for Extracting Microservices from Monolithic Enterprise Systems. Brazilian Workshop on Software Visualization, Evolution and Maintenance (VEM).
7	Balalaie, A., Heydarnoori, A., and Jamshidi, P. (2015). Migrating to Cloud-Native Architectures Using Microservices: An Experience Report. CloudWays Workshop.
8	Ouertani, S. (2015). From Microservices to SOA. Service Technology Magazine.
9	Rahman, M., and Gao, J. (2015). A Reusable Automated Acceptance Testing Architecture for Microservices in Behavior-Driven Development. IEEE Symp on Service-Oriented System Engineering (SOSE), pp. 321-325.
10	Nycander, P. (2015). Learning-Based Testing of Microservices. Workshop on Automating Test case design, Selection and Evaluation.
11	Namiot, D., Sneps-Sneppe, M. (2015) On Micro-Services Architecture. Intl Jrnl of Open Information Technologies
12	Microservice-based Architecture for the NRDC. International Conference on Industrial Informatics (INDIN), 2015
13	Savchenko, D. I., Radchenko, G. I., and Taipale, O. (2015). Microservices validation: Mjolnirr platform case study. Intl Conf on Cloud Computing and Services Science.
14	Versteden, A., Pauwels, E., and Papantoniou, A. (2015). An Ecosystem of User-facing Microservices supported by Semantic Models. 5th International USEWOD Workshop: Using the Web in the Age of Data.
15	Toffetti, G., Brunner, S., Blchlinger, M., Dudouet, F., and Edmonds, A. (2015). An architecture for self-managing microservices. Proceedings 1st Intl Workshop on Automated Incident Management in Cloud (pp. 19-24). ACM.
16	Krause, L. (2014). Microservices: Patterns and Applications.
17	Kukade, P. P., and Kale, G. (2015). Auto-Scaling of Micro-Services Using Containerization. International Journal of Science and Research.
18	Viennot, N., Lcuyer, M., Bell, J., Geambasu, R., and Nieh, J. (2015). Synapse: a microservices architecture for heterogeneous-database web applications. Proceedings 10th European Conference on Computer Systems.
19	Kratzke, N. (2015). About Microservices, Containers and their Underestimated Impact on Network Performance. Cloud Computing Conf. 2015.
20	Thones, J. (2015). Microservices. IEEE Software, 32(1).
21	Newman, S. (2015). Building Microservices. O'Reilly Media, Inc.

microservices approaches that helps us to taxonomically classify and compare the primary studies.

4.1 Microservice Reference Model

While we do not include blogs in the study as there is no formal, independent quality control in place, we use this section to extract an industry perspective on microservices from these.

Principles. A Microservice Architecture is a way of architecting software applications as independently deployable services. Based on (Lewis and Fowler, 2014), microservices can be characterise through a number of principles:

- organization around business capability.
- evolutionary design.
- deployment / infrastructure automation.
- intelligence in the endpoints.
- heterogeneity and decentralized control
- decentralized control of data.
- design for failure.

Except the first two items, the other point directly relate to the platform on which the architecture is deployed on – the cloud in our context.

Patterns. Based on these principles, patterns emerge to compose microservices. Arun Gupta proposes a number of patterns in his blogs[1]. Recommended patterns on how to compose microservices together

- Aggregator Microservice Design Pattern – e.g., a service invoking others to retrieve / process data.
- Proxy Microservice Design Pattern – a variation of the Aggregator with no aggregation.
- Chained Microservice Design Pattern – produces a single consolidated response to a request.
- Branch Microservice Design Pattern – extends the Aggregator and allows simultaneous re-

[1] http://blog.arungupta.me/microservice-design-patterns/.

sponse processing from possibly mutually exclusive chains of microservices.

- Shared Data Microservice Design Pattern – towards autonomy through full-stack services with control of all components.
- Asynchronous Messaging Microservice Design Pattern – use message queues instead of REST request/response pattern.

Looking at blogs[2] shows that coupling vs. autonomy in microservices is an open question on what messaging patterns to choose for microservices. Interaction pattern, e.g., Request-Reply vs. Publish-Subscribe are also discussed, as are event vs. command/query-based information exchange. Putting these latter two dimensions in a matrix would result in four options to realise couplings between microservices. Microservices are important for clouds and software architecture for the cloud as the cloud can ideally support microservices through services at different layers in different formats and the cloud can provide mechanisms to deal with the inherent challenges (uncertainty caused by heterogeneity, multi-tenancy etc).

4.2 A Characterisation Framework

We propose a classification framework that we will use to categorise the primary studies. This frameworks uses common terms from software engineering methods with methodological support, architecture, tool support and applications. The subheadings were identified after a first-round scans of selected studies in order to ensure the validity of the framework.

- *Methodological support*
  - Design – architecture-level software design
  - Testing – quality assurance / test methods
  - Configuration and management – deployment and quality related (e.g. scalability)
  - Migration – refactor/re-engineer legacy into microservices
  - Microservice identification – identify microservices in existing solutions
- *Architectural support* – software architecture
  - reference architectures
  - architectural modelling and specification
- *Platform/tool support*
  - Testing – and test beds and other tool support
  - Deployment/Provisioning platform – container repositories and engines
  - Microservice identification – tool support
- *Applications* – application domains/sectors
  - Types: e.g., Web apps, (smart) grids
  - Domains: e.g., financial services, transport

[2]Such as the blog https://www.voxxed.com/blog/2015/04/ coupling- versus-autonomy-in-microservices/

5 RESULTS

We categorise of the results 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 discussion is addressed.

5.1 Overview of the Primary Studies

In order to examine the state of research on microservices, the following questions are considered:

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

1) Temporal overview of studies. With only a few studies in 2014, there has been a dramatic increase in 2015, signaling a significant concern (Fig. 2). No relevant publications were found prior to 2014, as the terms has not been used as understood today.

2) Publication fora and formats. We have categorised the publication fora into the computing fields as follows (Figure 3):

- software engineering
- service engineering
- cloud computing
- networks, telecomms and distributed systems
- wider computing/IT/science context

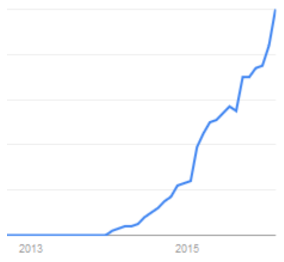


Figure 2: Trend Graph for Microservices.

Regarding the sources, we recognise and distinguish the following publication formats: journal [11,22], magazines [8,20], conferences and workshops [3,4,5,6,7,10,12,13,14,15,17,18,19] and theses [1,2,4] and books [16,21].

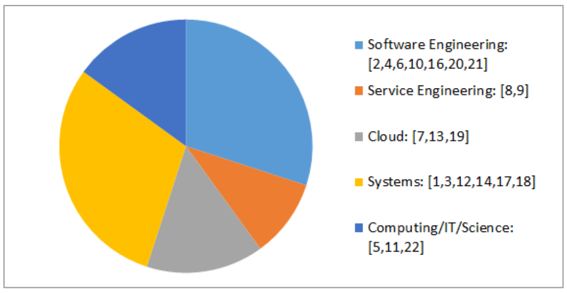


Figure 3: Study Distribution by Field.

3) Research methods for microservices. Typically, the contribution type (Solution Proposal, Evaluation Research, Validation Research, Experience Report, Review) and the evaluation method (Case Study, Mathematical Proof, Experience Report, Example Application, Controlled Experiment) are distinguished.

In Figure 4, the primary studies are organised according to their contribution type. Given the relative immaturity of the domain, the evaluations lack detailed experience reports and proofs, while equally controlled experiments, sample implementation as experience reports and some controlled experiments have been reported. Figure 5 complements this by looking more specifically at the technical contribution (categorising non-solution studies such as experience reports, evaluations and reviews as 'other'). More interesting here is the 3:2 split between an architecture perspective and a method perspective.

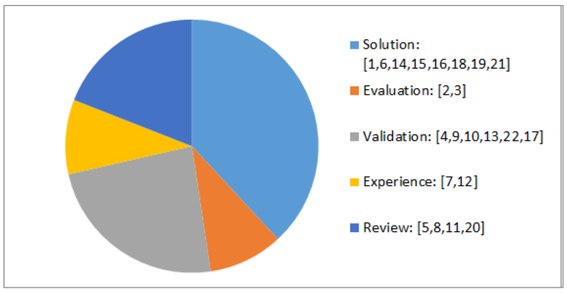


Figure 4: Study Distribution by Contribution Type.

Table 7 lists extracted keywords by frequency and Table 8 compares studies based on the core categories.

5.2 Discussion

The key terms extracted from all selected studies help to categorise them in terms of individual focus and

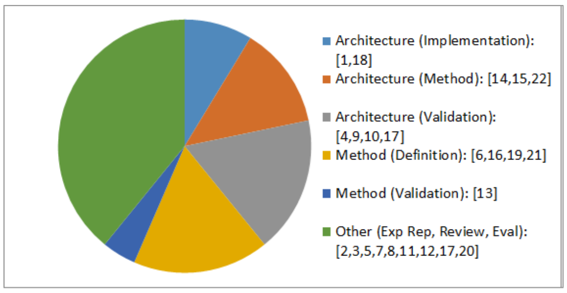


Figure 5: Study Distribution by Technical Contribution.

contribution (Figures 3 to 5) and to obtain an understanding of key research concerns. In order to address the latter concern, Table 7 shows the terms extracted:

- these have been categorised by following a common upper ontology (Pease et al., 2002) with 3 central concepts object (computational entity, activity (purpose) and property (quality), which has been supplemented by additional dimensions technology, research approach, application context and application.
- for each term, we recorded the number of occurrences – higher numbers of occurrences indicate a stronger research concern and/or stronger consensus on the importance of the aspect.

Some observations can be deduced from the key terms distribution and frequency. In terms of *key benefits* that characterise microservices, we can observe:

- **Architecture:** Microservice architectures are distributed. Microservices are component/service style entities, specifically characterised by being independently deployable and scalable in distributed architectures as primary concerns.
- **Method/Process:** Furthermore, maintainability and evolvability are further characteristics. Microservices need to be seen in a wider continuous development context (e.g., DevOps).
- **Deployment/Operation:** a third import concern emerges that refers to the deployment of microservices. While we limited the study selection to those clearly focusing on microservices (by selecting those with 'microservice' in the title), nonetheless, the key term prove the importance of the cloud and containers as independent deployment, autoscaling, Docker, container are very frequently mentioned.

There is agreement that this is a new architectural style. Microservices have been emerging out of services, but the link to containers is obvious. While at the deployment level, there is a clear focus on cloud, and container technologies in particular, the current

Table 7: Key terms extracted after two rounds of extraction and keyword frequency.

Q Quality	#	E Entity	#	P Activity	#	T Technology	#	C Context	#	R Research Approach	#	A Application	#
scalability	6	architectural style	5	testing	6	Docker	3	Monolithic Enterpr Syst	4	Experiment	2	Financial services	3
independently deployable	6	pattern	3	self-manage	5	SerfNode	1	IoT	1	Case Study	2	Transport	1
maintainable	3	Container	3	distribution	4	Spring	1	SOA	1	Analysis/ Comparison	1	Learning Technology	1
interoperable	2	smart grid	2	componentise	3			Libraries	1				
reusable	2	Web app	2	semantic modelling	3			Middleware	1				
user-facing	2	architecture	2	migration	3			Data Centre	1				
performance	2	network	1	auto-scale	2								
context-based	1	arch pattern	1	development	1								
heterogeneous	1	architecture paradigm	1	validation	1								
mobile	1			adoption	1								
reliable	1			packaging	1								
smart	1			replication	1								
flexibility	1			security	1								
				monitoring	1								

application domains are more traditional. Here, financial services (in a hybrid on-premise/private cloud scenario) and transport (more an edge cloud scenarios with the need to support lightweight virtualisation on smaller devices/sensors) have been investigated.

Another important perspective is the *application* of microservices in a long-term software evolution and modernisation context: (i) Microservices are often discussed in software and IT systems migration. (ii) Their SOA inheritance emphasises their suitability for modernising monolithic legacy systems.

The *forums* in which microservices are discussed are – in this order – the following: software engineering, service engineering, cloud computing, followed by others such as networks, operating systems and distributed systems. This demonstrates the nature of microservices as a service-oriented architectural style. Most of these include some reference to cloud computing, even if not published in this forum.

5.3 Classification of Microservice Methods and Techniques

In Table 8, we compared the studies based on the core classification categories. As already stated, microservices are considered as an architectural style, in which we can distinguish two technical perspectives (see column 'Technical Contribution, which refines the 'Contribution Type'):

- Architecture: architecture implementation and validation, but also architecture design methods.
- Methods: a more process-centric view with method definitions and validations.

Figure 6, where we mapped the generic contribution types (Solution, Evaluation, Experience Report, Review) to more specific technical contributions relevant in the microservices context (Architecture and Method), makes the distribution of technical contributions more clear. The bubble size indicates the number of studies of that type.

Solutions and their experimental validation dominate (the 2 left-most columns Sol and Val). We can also see that Architecture is more prevalent (the lower 3 rows AI, AM, AV) than the process-centric Method perspective (the 2 rows MD and MV above) – see Figure caption for acronym definition. In the upper right corner, non-solution/validation studies are summarised. Here systematic evaluations and experience reports on the one hand (Eval, Exp) and technology reviews (Rev) are equally frequent.

5.4 Threats to Validity

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

Threats to the Identification of Primary Studies. A challenge was to determine the scope of our study, since microservices relate to different computing and IT communities including software engineering, information systems and cloud. These communities use different terminologies for the same concepts. To cover all and avoid bias, we searched for the microservice 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.

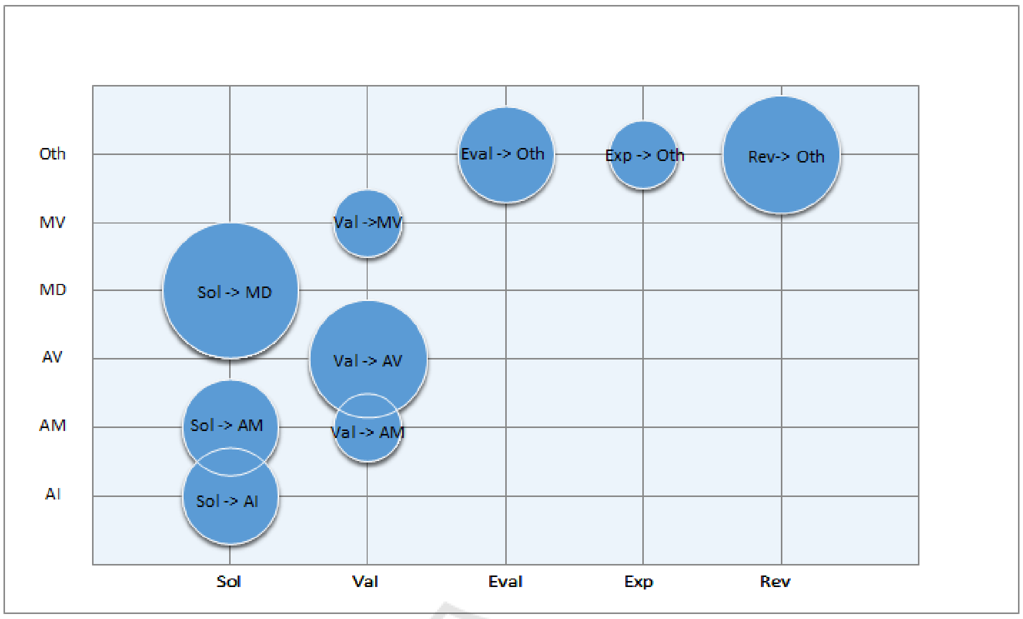


Figure 6: Study Distribution by Technical Contribution over Contribution Type (bubble plot). [Sol:Solution, Val:Validation, Eval:Evaluation, Exp: Experience Report, Rev: Review; AI:Architecture Implementation, AM:Architecture Method, AV:Architecture Validation, MD:Method Definition, MV:Method Validation, Oth:Others] .

Threats to Selection and Data Extraction Consistency. The formulation of the research questions has helped in selecting studies of relevance, as did the Reference Model and Characterisation Framework in Section 3. However, we did include magazine contributions and thesis here (as along as a review took place) to include all trends and activities.

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

6 RESEARCH IMPLICATIONS AND FUTURE DIRECTIONS

While the maturity is low, valuable insights into trends can be identified that are of benefit to researchers and practitioners alike.

6.1 Maturity

The actual scientific *contributions* are a mix of technology reviews, test environments and use case architectures (conceptual and implemented). As a good part of this is still conceptual, it can be seen as a sign of immaturity. This interpretation is strengthened by the fact that some use case validations and no larger-scale empirical evaluations exist.

Regarding the contribution formats, there is also a notice imbalance compared to more mature domains:

- Larger number of thesis (BSc/MSc level) and magazine article – pointing at an emerging topic through initial experimental and explorative work particularly at Bachelor or Master level, and also in magazines providing early technology reviews. The number of journal publications is low (with short communications published only).
- Higher number of use cases in comparison with technology solutions – again pointing at an emerging topic by aiming to formatively validate the technology through use cases, rather than a more systematic summative evaluation.

We can note specifically a lack of proven technologies to realize a microservices architecture, understanding to differentiate SOA from microservices, monitoring tools for microservices, architectural pattern for microservices, experimental approaches to empirically compare microservices with other styles (for example using architecture evaluation mathods like ATAM), tools to enable performance-driven DevOps for microservice architecture development, software enigneering methods (methodologies, design patterns, best practices, quality assurance, system versioning, change management) for microservice architecture development, and successful/unsuccessful examples of microservices development (only a few exist, c.f. Netflix).

Table 8: Study comparison in terms of characterisation framework (focussing on architecture and method support – column Technical Contribution).

ID	Title	Format	Field	Contrib Type	Technical Contribution	Contribution Details
1	Distribution of microservices for hardware interoperability in the Smart Grid	Thesis	Telecoms	Sol	Architecture Implement	microservice-based smart grid impl
2	Evaluation of Microservices as an Extension of Component Technologies (in German)	Thesis	Soft Eng	Eval	Concept Evaluation	conceptual evaluation
3	Location and Context-Based Microservices for Mobile and Internet of Things Workloads	Conf/Worksh	Mobile Comp	Eval	Concept Evaluation	3 sample use cases (microserv)
4	Learning-Based Testing of Microservices: An Exploratory Case Study Using LBTest	Thesis	Soft Eng	Val	Architecture Validation	microservice test bed
5	Distributed Systems of Microservices Using Docker and Serfnode	Conf/Worksh	Science	Rev	Review	review of container tech
6	Towards a Technique for Extracting Microservices from Monolithic Enterprise Systems	Conf/Worksh	Soft Eng	Sol	Service	technique for ms identification in enterprise arch
7	Migrating to Cloud-Native Architectures Using Microservices: An Experience Report	Conf/Worksh	Cloud	Exp	Experience Report	migration experience report
8	From Microservices to SOA	Magazine	Services	Rev	Review	technology review
9	A Reusable Automated Acceptance Testing Architecture for Microservices in Behavior-	Conf/Worksh	Services	Val	Architecture Validation	testing architecture for ms
10	Learning-Based Testing of Microservices	Conf/Worksh	Soft Eng	Val	Architect Validation	microservice test bed
11	On Micro-Services Architecture	Journal	Science/ Engineer	Rev	Review	technology review
12	Microservice-based Architecture for the NRDC	Conf/Worksh	Systems	Exp	Experience Report	data centre architecture using microservices
13	Microservices validation: Mjolnirr platform case study	Conf/Worksh	Cloud	Val	Method Validation	testing for ms review + validation method for ms
14	An Ecosystem of User-facing Microservices supported by Semantic Models	Conf/Worksh	Web/ Semantics	Sol	Architecture Method	abstract architecture for ms for Web apps
15	An architecture for self-managing microservices	Conf/Worksh	Cloud	Sol	Architect Method	(concept) architecture for self-management of microservices
16	Microservices: Patterns and Applications	Book	Soft Eng	Sol	Method Definition	design methods and patterns
17	Microservice-based Architecture for the NRDC	Conf/Worksh	Systems	Exp	Experience Report	data centre architecture using microservices
18	Synapse: a microservices architecture for heterogeneous-database web applications	Conf/Worksh	Systems	Sol	Architecture Implement	(implement) architecture for microservices for web DB apps
19	About Microservices, Containers and their Underestimated Impact on Network Performance	Conf/Worksh	Cloud	Sol	Method Definition	analysis and design recommendations
20	Microservices	Magazine	Soft Eng	Rev	Review	technology review
21	Building Microservices	Book	Soft Eng	Sol	Method Definition	full lifecycle methodology (design, testing, impl/devops)
21	Auto-Scaling of Micro-Services Using Containerization	Journal	Science/ Engineer	Val	Architecture Method	conceptual architecture and scaling approach
23	A Microservice Approach for Near Real-Time Collaborative 3D Objects Annotation	Conf/Worksh	Web/ Semantics	Val	Architecture Validation	microservices use case in learning technology

6.2 Research Trends

In terms of a perceived need for research, the following aspects are important regarding *methodological and tool support*:

- Testing is of particular importance (microservices as design artefact towards satisfying an integration need)
- Intelligent semantic technologies for their management for instance for discovery in repositories.
- Resulting from independence, self-manageability enabled by the deployment platform.

Further trends can be added: microservices migration, autonomous healing, runtime architecture change, architectural patterns, DevOps and microservices, microservice monitoring, auto-scaling in the context of microservices, configuration optimization for microservices, and architectural refactoring.

6.3 Benefits for Researchers and Practitioners

The characterization framework provides a holistic view of different microservices concerns to be considered. The classification and comparison of studies, which contains overall 6 comparison attributes presented across the figures and tables, provides useful information. For the 21 papers and 6 comparison attributes, it creates a collection with 21*6 = 126 data points. This is beneficial for researchers who require a quick identification of relevant studies and detailed insight into state-of-the-art that supports microservices, but also for practitioners interested in understanding existing methods, architectures and tools for microservice development and deployment.

7 CONCLUSIONS

Microservices have only received attention very recently (Lewis and Fowler, 2014; Newman, 2015), driven by two factors. Firstly, they address limitations of the SOA style (Erl, 2005), specifically linking it to independent deployability and lightweightness. Companies such as Netflix and Thoughworks have been at the forefront of this.

This brings this discussion also into the context of continuous development approaches (Fitzgerald and Stol, 2014) such as DevOps (Brunnert et al., 2015). Furthermore, cloud technology (Mell and Grance, 2011; Antonopoulos and Gillam, 2010) and container technology in this context in particular (Pahl and Lee, 2015; Pahl, 2015) provide a mechanism to deploy microservices consistent with the style principles. Microservice patterns need to be mapped onto cloud patterns (Pahl and Jamshidi, 2015; Fehling et al., 2014).

While the maturity of the research work is quite low, given the recent emergence of the topic, a conclusive summative analysis is not possible, but good pointers towards research gaps and directions can be derived that can be seen as a contribution of a more formative investigation of the domain.

In conclusion, from our mapping study, microservices emerge as an architectural style, but one that extends from the 'design-stage architecture' into deployment and operations as a continuous development style – the 'method' dimension. It also seem from a significant part of the studies reviewed to be intrinsicly linked to cloud-based containers for deployment and dynamic management - the 'dynamic architecture' dimension.

REFERENCES

Antonopoulos, N. and Gillam, L. (2010). *Cloud computing: Principles, systems and applications*. Springer.

Brunnert et al., A. (2015). Performance-oriented devops: A research agenda. *Technical Report: SPEC-RG-2015-01, SPEC RG DevOps Performance Working Group*.

Erl, T. (2005). *Service-oriented architecture: concepts, technology, and design*. Pearson Education.

Fehling, C., Leymann, F., Retter, R., Schupeck, W., and Arbitter, P. (2014). Cloud computing patterns.

Fitzgerald, B. and Stol, K.-J. (2014). Continuous software engineering and beyond: Trends and challenges. In *1st Intl Workshop on Rapid Continuous Software Engineering*, RCoSE 2014, pages 1–9.

Jamshidi, P., Ahmad, A., and Pahl, C. (2013a). Cloud migration research: a systematic review. *Cloud Computing, IEEE Transactions on*, 1(2):142–157.

Jamshidi, P., Ghafari, M., Ahmad, A., and Pahl, C. (2013b). A framework for classifying and comparing architecture-centric software evolution research. In *7th European Conference on Software Maintenance and Reengineering (CSMR), 2013*, pages 305–314.

Kitchenham, B., Brereton, O. P., Budgen, D., Turner, M., Bailey, J., and Linkman, S. (2009). Systematic literature reviews in software engineering–a systematic literature review. *Information and software technology*, 51(1):7–15.

Lewis, J. and Fowler, M. (2014). *Microservices*. http://martinfowler.com/articles/microservices.html.

Mell, P. and Grance, T. (2011). The NIST definition of cloud computing.

Newman, S. (2015). *Building Microservices*. O'Reilly.

Pahl, C. (2015). Containerization and the paas cloud. *IEEE Cloud Computing*, 2(3):24–31.

Pahl, C. and Jamshidi, P. (2015). Software architecture for the cloud - a roadmap towards control-theoretic, model-based cloud architecture. In *Europ Conference on Software Architecture ECSA'15*.

Pahl, C. and Lee, B. (2015). Containers and clusters for edge cloud architectures - a technology review. In *3rd International Conference on Future Internet of Things and Cloud (FiCloud-2015)*. IEEE.

Pease, A., Niles, I., and Li, J. (2002). The suggested upper merged ontology: A large ontology for the semantic web and its applications. In *AAAI-2002 workshop on ontologies and the semantic web*.

Petersen, K., Feldt, R., Mujtaba, S., and Mattsson, M. (2008). Systematic mapping studies in software engineering. In *12th Intl Conference on Evaluation and Assessment in Software Engineering*, volume 17.

Zimmermann, O. (2009). *An architectural decision modeling framework for service oriented architecture design*. PhD thesis, Universität Stuttgart.