Cloud topology and orchestration using TOSCA: A systematic literature review

(Technical Report*)

Julian Bellendorf and Zoltán Ádám Mann

paluno – The Ruhr Institute for Software Technology
University of Duisburg-Essen, Essen, Germany
{firstname.lastname}@paluno.uni-due.de

Abstract. Topology and Orchestration Specification for Cloud Applications (TOSCA) has become the de facto standard for specifying the topology of cloud applications, their deployment on cloud resources, and their orchestration. In recent years, also the cloud research community has shown increasing interest in TOSCA, leading to an increasing number of publications. These publications address different topics around TOSCA, e.g., devise sophisticated cloud orchestration methodologies using TOSCA, extend the language of TOSCA, or present tools for manipulating TOSCA models. To help researchers and practitioners overview this multifaceted area of research, this paper presents the results of a systematic survey of the relevant literature.

Keywords: TOSCA · cloud · topology · orchestration.

1 Introduction

With the widespread adoption of cloud computing, more and more applications are deployed in a cloud setting. For complex applications, comprising many components with different technical dependencies and constraints, also making use of different platform components, the deployment and ongoing management tasks have become dauntingly complicated and error-prone using traditional tools [59]. Also, the lack of interoperability between different tools became a severe limitation [8]. Thus, the need arose to describe cloud applications and related management tasks on a higher level of abstraction, in a standardized format.

To address these issues, the Organization for the Advancement of Structured Information Standards (OASIS) published the Topology and Orchestration Specification for Cloud Applications (TOSCA) standard in 2013 [65]. TOSCA is a modeling language addressing the deployment and portability of applications, as well as the interoperability and reusability of individual components of these applications [6]. The concept of TOSCA includes two aspects: (i) the structural

* Abridged version accepted for the 7th European Conference on Service-Oriented and Cloud Computing (ESOCC 2018)

description of a composite cloud application as a *topology graph* and (ii) *management plans* for deploying and maintaining cloud applications.

The description of the topology of an application uses two types of elements: nodes and relationships. Nodes represent individual components, and also define management operations for those components, such as for creating, configuring or starting the component [43]. Relationship types describe the relationships of the individual components to each other; for example, a "hosted-on" relationship can be used to allocate virtual resources to the respective physical components.

TOSCA foresees two types of processing for the deployment of composite applications [10]. In *imperative processing*, a Management Plan defines the exact management operations and their execution order. TOSCA uses existing workflow languages such as Business Process Model and Notation [66] and Business Process Execution Language [64] for the definition of Management Plans. For *declarative processing*, no Management Plans are defined; instead, a runtime system interprets the application topology and infers the necessary steps for typical operations (e.g., deployment) based on appropriate conventions.

The original TOSCA specification was based on XML. The simplified TOSCA profile, released in 2016, used YAML, "to simplify the authoring of TOSCA service templates" [67]. For a more detailed introduction to TOSCA, the reader is referred to the overview papers of Binz et al. [6] and Brogi et al. [25].

Beside its industrial adoption[1], TOSCA also sparked significant interest in the research community. It has played various roles in different research approaches: some used TOSCA as part of a more general methodology, others extended the modeling capabilities of TOSCA, or designed tools to support the manipulation of TOSCA models. This multifaceted use of TOSCA and the growing number of relevant papers make it hard to track all related research. Therefore, the aim of this paper is to give an overview about the use of TOSCA in the research community. We performed a systematic literature survey to devise a taxonomy of the main research topics that have been addressed in connection with TOSCA in recent years. The analysis of the results also gives hints about potential directions for future research.

2 Survey methodology

For our literature survey, we used a systematic methodology [16]. Fig. 1 gives an overview of the process. In the first step, we used keyword search to identify the set of possibly relevant papers. For this purpose, we used the Scopus database[2] and applied the following search string:

> "Topology and Orchestration Specification for Cloud Applications" AND
> (ABS (tosca) OR TITLE (tosca))

Here, ABS and TITLE mean that the given expressions must be contained in the abstract respectively title of the paper. Searching for the keyword TOSCA in

[1] See, e.g., https://wiki.oasis-open.org/tosca/TOSCA-implementations
[2] https://www.scopus.com

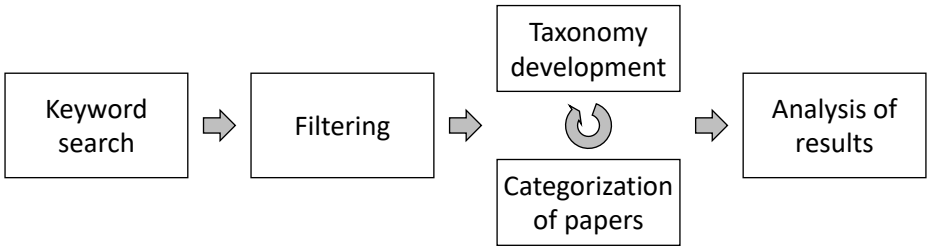

Fig. 1. Overview of survey methodology

the abstract or the title ensures that TOSCA is a main aspect of the respective paper. In addition, we are looking for the full version of the term (Topology and Orchestration Specification for Cloud Applications) to exclude papers that use the word TOSCA in another meaning.

We focused on the period from 2012 to 2017 (also including the years 2012 and 2017). 2012 was the year in which the first TOSCA papers were published. We considered only whole years and thus did not consider papers that were published after 2017. In addition, we excluded very short papers (less than 4 pages). We found 89 papers with this search and sorted out 6 of them due to their length. Then we used Google Scholar[3] to find more papers that could not be found in the Scopus database, but fit the described search string. In this way 8 more papers could be found, leading to a total of 91 papers.

Afterwards, we read each paper and categorized it using open coding. Parallel to the processing of the papers, we were continuously refining our coding scheme, building up a taxonomy in a bottom-up fashion, and also refining the categorization of already processed papers. At the end of this phase, the taxonomy was finished and each paper categorized according to this taxonomy. In the last step, we analyzed the results to identify focal points of existing research, trends, as well as possible directions for further research.

3 Survey results

TOSCA was officially introduced in 2013, but the first papers about TOSCA were published already in 2012. As Fig. 2 shows, the number of papers quickly rose to reach a peak in 2014. In the subsequent two years, the number of publications decreased, but 2017 marked a significant growth again, actually leading to the highest number of papers in a single year. This renewed interest may be related to the release of the simplified TOSCA profile in 2016.

Fig. 3 presents the taxonomy that we developed based on the analyzed papers. On the highest level, we categorized the papers based on their *main contribution regarding TOSCA*. We identified the following categories:

[3] https://scholar.google.com

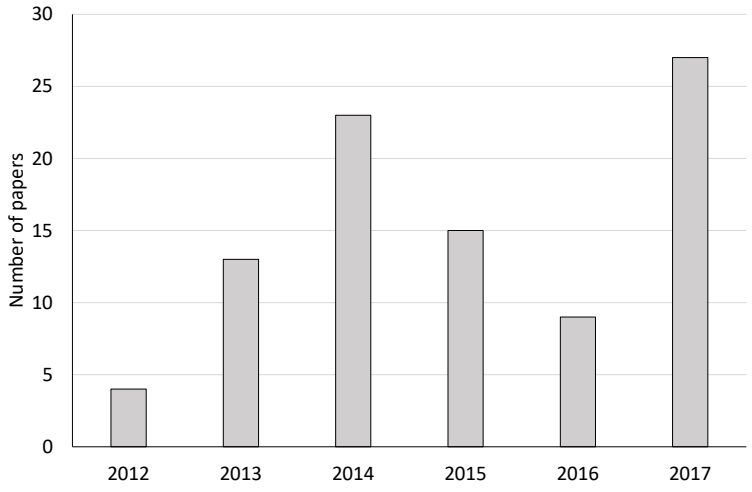

Fig. 2. Evolution of the number of papers over time

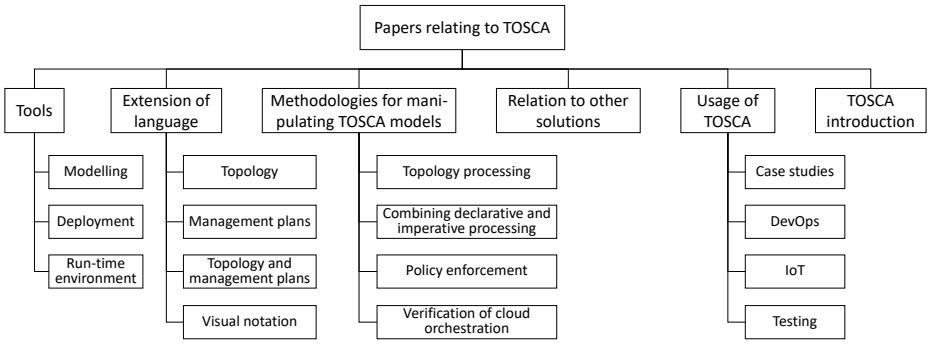

Fig. 3. Taxonomy for categorizing the processed papers

- **Tools**: papers describing a tool for TOSCA. Further categorization is possible based on the type of tool; in particular, this includes modeling tools, tools for deployment automation, and run-time environments.
- **Extension of language**: approaches that extend the TOSCA language. The extension may relate to topologies, Management Plans, or both. Furthermore, extensions aiming at a visual notation also belong to this category.
- **Methodologies for manipulating TOSCA models**: papers describing a methodology for processing TOSCA models. The identified sub-categories are: papers about processing TOSCA topology models, approaches that combine declarative and imperative processing, solutions to enforce some given policies, and verification of cloud orchestration.
- **Relation to other solutions**: papers about a relation (e.g., comparison or conversion) between TOSCA and some other technique.

- **Usage of TOSCA**: papers demonstrating the use of TOSCA. This includes case studies that showcase the practical applicability of TOSCA, papers about the contribution of TOSCA to DevOps, papers about the use of TOSCA in an IoT (Internet of Things) setting, as well as papers about the use of TOSCA models for testing purposes.
- **TOSCA introduction**: papers that introduce TOSCA or some of the concepts within TOSCA.

The following subsections describe some representative papers in each category of the taxonomy of Fig. 3, except for the category "TOSCA introduction." (Due to space limitation we cannot describe all found papers in detail.)

3.1 Tools

The Tools category consists of papers mainly presenting tools for describing, deploying, and instantiating cloud applications using TOSCA. Prototypes that only serve to evaluate an approach are not assigned to this category.

Kopp et al. present the web-based modeling tool Winery [52]. First, Winery includes the Topology Modeler, with which components can be combined to form an application topology. Second, Winery contains the Element Manager, which can be used to create, modify, and delete the components contained in the topologies. In addition to Topology Modeler and Element Manager, Winery includes a repository to store the data for the other two components. Kopp et al. also propose an extension to Winery that can be used to model Management Plans [53]. However, this approach was only implemented as a prototype.

Binz et al. describe OpenTOSCA, a runtime for imperative processing of TOSCA applications [5]. This runtime serves to execute the defined Management Plans respectively the operations described within the nodes. Wettinger et al. present an extension to OpenTOSCA in the form of a unified invocation interface [92, 93]. This interface provides the logic to perform management operations and hides the technical details of the underlying technologies. The interface is called from the Management Plans to provision an application.

Breitenbücher et al. add Vinothek to these two tools [12]. Vinothek offers the user an interface for providing an instance of an application. For this purpose, the user is offered the set of applications without having to deal with the technical details. The selected application is started by a connected runtime.

Katsaros et al. also provide a tool to deploy and manage software components [48]. The execution environment TOSCA2Chef parses TOSCA documents and deploys the components described in OpenStack Clouds using the Opscode Chef configuration management software and BPEL processes.

3.2 Extension of language

This subsection deals with the extension of the language of TOSCA. This extension can affect the Topology elements defined in the standard and/or the languages used by TOSCA to describe the Management Plans.

Brogi et al. extend TOSCA by means for specifying the behavior of the application components when executing the management operations defined in the nodes [17–19]. Considering the effects of the operations to be performed and the states that the components assume after execution, the validation of Management Plans is possible. In a valid Management Plan, all actions are performed in the correct order. In an invalid plan, for example, the start process could be defined to run before installing a component.

Zimmermann et al. define an extension to the topology modeling in TOSCA, regarding the business operations of the individual components, which should be displayed in a technology-agnostic manner [99]. The application deployment described in this approach includes a service bus for communication between the individual components, respectively for calling these business operations using identifiers and interface descriptions.

Other papers focus on expanding the language of the Management Plans. Kopp et al. extend the Business Process Model and Notation (BPMN) to provide direct access to the topology elements [51]. The extension, called BPMN4TOSCA, can be transformed into standards-compliant BPMN. Vetter shows an approach that describes the Management Plans through XML nets [88]. This description allows the search for errors in the defined plans by finding anti-patterns. An anti-pattern could be for instance a missing or unneeded action.

Breitenbücher et al. propose a visual notation to unify the presentation of nodes within the topology, to avoid misunderstandings in the interpretation of the topologies defined in XML [13]. This involves the presentation of nodes and relationship templates, as well as groups for combining multiple elements into a single element to reduce the complexity of large representations.

3.3 Methodologies for manipulating TOSCA models

Various approaches have been proposed to work with TOSCA models. Some focus on the processing of topologies, whereas others also take Management Plans into account to enforce policies, to verify the orchestration of the components, or to combine declarative and imperative processing.

Processing of Topologies Brogi and Soldani describe an approach that involves matching between individual Node Types and Service Templates [22–24]. This matching allows sets of Node Types to be grouped together in a topology to reduce its complexity. In addition, proven combinations of Node Types can be reused in new application topologies. For this purpose, the authors define *(i) exact matching* between Node Type and Service Template and propose further definitions for *(ii) plug-in*, *(iii) flexible* and *(iv) white-box matching*. *Exact matching* between Service Template and Node Type exists if the requirements and offered services are perfectly compatible with each other. In the case of *plug-in matching*, the requirements of the Service Template are weaker and the capabilities offered higher than those of the Node Type. The other two types, *flexible* and *white-box matching*, involve non-essential, syntactic differences between feature names and missing Service Template requirements or capabilities.

Service Templates with the other three properties can be adapted to create a new template that fits exactly a given node type.

Binz et al. propose sharing resources to save costs [7]. The rationale is that container components (for example, virtual machines) are underutilized by one single application component and additional components can be hosted by these containers. For this purpose, the topologies from two applications that use the same container components are merged into a topology after a feasibility check has been performed. This process results in a topology in which both applications retain their respective functionality. Saatkamp et al. present another approach to save costs when running cloud applications, by adapting the application topology when a provider specifies a new offer and certain components of the application need to be migrated to new container components [76]. The approach checks whether the capabilities offered match the requirements of the given application components and splits the topology over the providers' offerings.

Automatic completion of TOSCA topology models has also been considered. The aim is that an application developer only has to model the business-relevant components and the underlying infrastructure is automatically added. This leads to a reduction of the effort because the developer does not have to know the technical details of the infrastructure, platform and software components to deploy an application. Multiple approaches were proposed to achieve this aim. The approach of Hirmer et al. is based on a repository of nodes and relationships [45]. This repository is used by a prototypical extension of Winery to complete incomplete topologies. The developer can follow the completion of the topology step by step and adjust if necessary. The approach of Panarello et al. also tracks the completion of application topologies [68]. This involves distributing the components of an application through the offerings of the providers of a federation. A Decision-Making component selects the appropriate offers and a Topology Completion Manager completes the incomplete topology accordingly. Also Brogi et al. present an approach for a related purpose [21]. It uses the prototype DrACO, which receives information about suitable cloud offerings through crawling the network and storing their TOSCA representation in a repository. This representation can be used by the application developer to complete the topology.

Soldani et al. present TOSCAMart, an approach to reuse proven topologies in new environments [81]. TOSCAMart is based on a repository of various existing topologies provided to an application developer for the development of a new composite application. The developer defines a node in the new topology that describes the requirements for the fragment being inserted. TOSCAMart then selects a suitable solution for these requirements from the repository. Matching between a given node and the fragments from the repository is based on the definitions for *exact* and *plug-in* matching mentioned earlier.

TOSCA Node Types point to artifacts implementing the installation and deployment operations. Selecting an artifact that meets the requirements of the user requires technology-specific expertise. Combining artifacts to deploy complex applications is difficult and error-prone. To solve these problems, Endres et al. present a method that crawls open-source repositories and transforms the

found artifacts into technology-agnostic topology models [37]. These topology models facilitate the understanding of the functionality of the associated artifacts and can be combined to model the deployment of complex applications.

Combining declarative and imperative processing Breitenbücher et al. propose an approach that combines declarative and imperative to hybrid processing [10, 11]. Declarative processing is not suitable for larger applications because only general components and predefined operations known by the runtime can be executed. The disadvantage of imperative processing is that creating Management Plans is time-consuming, expensive, and error-prone. To overcome the drawbacks of both, the approaches are combined to a hybrid approach. The defined application topologies are interpreted and finally the associated Management Plans are generated. These can be finally adjusted by the developer.

Calcaterra et al. present a similar approach, also based on interpreting a topology and providing the appropriate Management Plan [26]. To do this, a converter translates the YAML model into a BPMN model, which is used to deploy the application. In addition, the approach still provides the ability for cloud providers to define and deliver their offerings so that users can choose the offer that best suits their needs. However, the presented prototype of this application only translates some of these functions.

Policy Enforcement Waizenegger et al. present a prototypical implementation of a TOSCA runtime extension to enforce policies describing non-functional requirements, specifically security properties, such as the encryption of a database or the geographic positioning of privacy-related data [89, 90]. Policies can be defined using both single-node management operations and Management Plans. Developers can create different instances of a policy so that the user can choose a specific version without having to adjust the entire template.

Verification of Cloud Orchestration Yoshida et al. describe an approach to the formal verification of TOSCA topologies that can be used to test the achievement of a particular target state in declarative processing of the TOSCA model [98]. The execution of the management operation is described by a state transition system in which a state with a certain property is to be reached.

Also Chareonsuk and Vatanawood deal with the formal verification of cloud orchestration design [31]. However, this approach, unlike that of Yoshida et al., considers imperative processing. For this purpose, a CSAR file containing the TOSCA topology and associated BPEL processes is converted into a formal model, to which a Model Checker is applied. The WSDL description of the used web services is also considered to take into account the signatures of the individual method invocations for verification. The authors use the description of safety properties as linear temporal logic formulae for model checking.

The approach of Tsigkanos and Kehrer is about defining patterns and anti-patterns and finally checking their presence or absence in the topology of a TOSCA service template, so that quality aspects can be proven [86].

3.4 Relation to other solutions

This category includes papers discussing (i) the comparison between TOSCA and other methods for describing cloud resources, and (ii) the automatic translation between TOSCA and another modeling language.

A comparison between TOSCA and the Heat Orchestration Template (HOT) is provided by Di Martino et al. [33]. HOT is the template format used to define the structure of an application for declarative processing by the OpenStack orchestrator Heat. The main difference between the two approaches is that HOT does not support workflow definition, which is possible with TOSCA: HOT is purely declarative, whereas TOSCA also supports imperative processing using Management Plans. Also similarities are shown, for example, both provide a catalog of nodes and resources that can be composed to applications.

For the translation of TOSCA into the HOT format, OpenStack has developed the project HEAT-Translator. This takes a format as input that is not compatible with HEAT and translates it into the HOT format. Among others Tricomi et al. present a deployment approach using the HEAT-Translator [85].

Yongsiriwit et al. address the interoperability of standards for describing cloud resources: TOSCA, Open Cloud Computing Interface (OCCI) and Cloud Infrastructure Management Interface (CIMI) [97]. For interoperability, ontologies are defined that describe the resources noted in each standard. In addition, an upper-level ontology is presented to describe cloud resources regardless of the used standards. Using inference rules, the special descriptions can be translated into this higher-level format and vice versa, which also allows the translation from one standard to another. Using the upper ontology, a knowledge base could be created, providing insights into relationships and possible inconsistencies.

3.5 Usage of TOSCA

This category consists of approaches that use the existing TOSCA notation. Some papers present a case study of the use of TOSCA. In some others, TOSCA is not the main focus, but only a means to achieve some goal. Some further papers discuss integrating TOSCA with other concepts to combine their benefits.

Kostoska et al. present a case study of the use of TOSCA for specifying the University Management System iKnow [54]. This system offers professors and students a platform to exchange electronic information and provide electronic services. The paper describes how the components of this application are defined using TOSCA. Examples show excerpts from the XML specification of node and relationship types, as well as artifacts that can be used for deployment and installation. It also mentions the challenges of using TOSCA for the specification of this application.

A different domain for using TOSCA is the specification of Internet of Things (IoT) applications. This area is addressed by various authors. Li et al. show how TOSCA can be used for an IoT application, namely an Air Handling Unit (AHU) that controls the condition and circulation of air in modern buildings [57]. The authors give examples of how to define the components of the application

and describe their experience using TOSCA for this use case. Also Da Silva et al. demonstrate the feasibility of defining IoT applications using TOSCA, in the context of different technologies [32]. In another paper, Da Silva et al. address the multitude of sensor data produced in IoT scenarios [79]. The authors describe how Complex Event Processing Systems can be deployed using TOSCA to process the incoming data and efficiently use network resources.

Wettinger et al. show the use of TOSCA in the DevOps area [94, 95]. A core principle of DevOps is to automate the deployment process to enable continuous software delivery. The authors address the challenge of combining different DevOps artifacts. For this purpose, a framework is presented to search public repositories for such artifacts. These artifacts are converted into TOSCA format to ensure a uniform representation and to enable their combination. Specifically, the transformation of Chef cookbooks and Juju charms is described. In another paper, Wettinger et al. integrate Configuration Management with Model-Driven Cloud Management in the context of DevOps [91]. Model-driven cloud management provides an overview of the structure of a complex application and supports the developer in handling necessary infrastructure changes, abstracting from lower-level actions (like installation or configuration of components), which in turn are provided by Configuration Management. The paper deals with the integration of TOSCA as a vehicle for Model-Driven Cloud Management and Chef as a Configuration Management tool. Integration aims at defining a holistic service model while ensuring its portability.

Cardoso et al. integrate TOSCA with the service description language USDL [27]. USDL is used to describe service offerings of an application, so that service discovery and selection can be simplified. TOSCA is used to deploy and manage the components of an application. This paper describes how the two languages can be combined so that parts of the lifecycle of an application can be automated.

Sampaio et al. define an approach to performing performance tests for cloud application topologies [78]. For the tests, different variants of a topology are specified with the help of TOSCA. After being deployed, these variants are evaluated on the basis of performance parameters chosen by the user.

4 Discussion

We analyzed the distribution of papers among the taxonomy categories, also taking into account the evolution along the time axis. Fig. 4 shows the distribution of papers among the categories for the two halves of the investigated time period (first respectively second column). It can be observed that the two biggest categories are “Usage of TOSCA” and “Methodologies for manipulating TOSCA models.” This holds for both halves of the period. Also, both categories grew significantly over time. The biggest relative growth, however, was produced by the “Relation to other solutions” category, which grew from 0 to 7. Also the “Extension of language” category grew considerably. A possible explanation is that such papers address rather specific and advanced topics that gain more relevance when the given field of study matures.

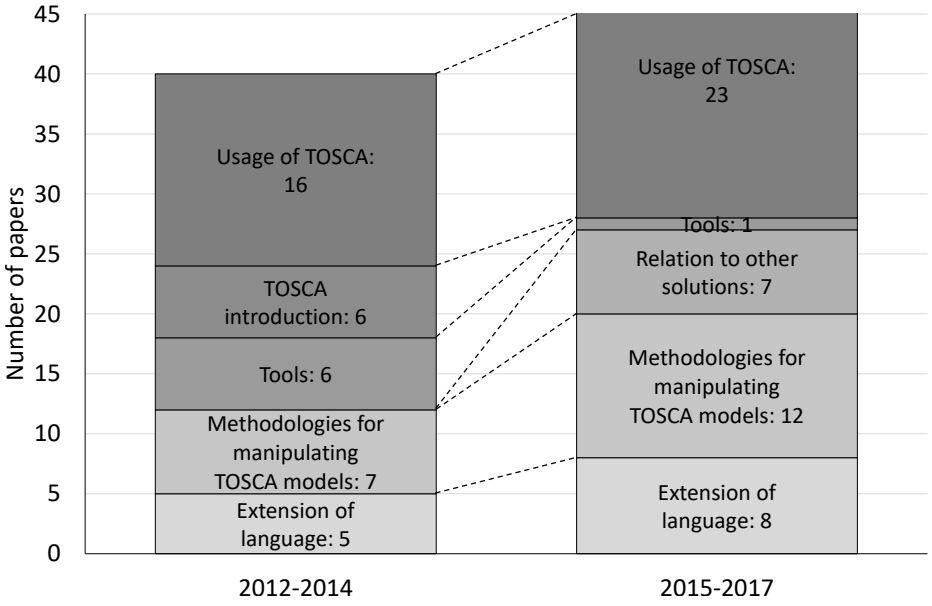

Fig. 4. Evolution of the number of papers belonging to the categories of the taxonomy

There were also two categories that diminished. Understandably, "TOSCA introduction" papers were only published in the first half of the period when TOSCA was new. More interestingly, also the number of "Tools" papers decreased considerably. This might be explained by the fact that the industrial adoption of TOSCA has led to the development of tools in industry, so that tool development has become less relevant for research.

Qualitatively, the findings of our survey show the *versatility* of TOSCA: its use in many different domains (also beyond cloud computing), for different purposes, in different phases of the service lifecycle, by different groups of users. This versatility is mainly due to (i) the possibility to define custom types for nodes, relationships, and capabilities and (ii) the possibility to define and manipulate partial topologies. Indeed, most surveyed papers use some additional types (and this is also planned for future research [61, 32]). Partial topologies make it possible to define an application without the technical details of the physical components and to optimize the deployment (which is also planned to be used in the future by several authors [68, 21, 79]). However, this versatility also poses the risk of the proliferation of different and incompatible TOSCA dialects. Hence we expect that *interoperability* will play an increasingly important role.

Furthermore, the survey results show that the following topics have received limited attention so far and represent important topics for future research:

– Given the enormous importance of *security* in cloud computing, it is striking that very few papers address it so far (although several authors mentioned

it as future work [78, 97, 76]). Also, TOSCA support for other related topics like *privacy* need to be investigated.
- The topic of *verification and validation* (V&V) is also addressed by few papers. Since V&V activities make up a significant part of the IT budget of an organization, we expect to see more work on how TOSCA can be used to make V&V more efficient and effective.
- Partial topologies open many possibilities for *optimization*, from which only a little has been investigated, mainly in connection with cost minimization. Many other aspects of optimization, e.g., related to performance and reliability, are yet to be explored.
- TOSCA has been shown to be useful in areas such as IoT and DevOps. We expect to see TOSCA being applied to new domains like *fog and edge computing* or *mobile computing*. TOSCA could be used as a framework for integrating different technologies, e.g. to support mobile cloud computing.

5 Conclusions

This paper presented a systematic literature review on TOSCA. The identified papers addressed a variety of topics, which we grouped into six categories (some further subdivided into several sub-categories). The categories with the highest number of papers were: (1) papers reporting on the use of TOSCA, (2) methodologies for manipulating TOSCA models, and (3) extensions of TOSCA.

Our survey showed the versatility of TOSCA, but also discovered areas that are hardly explored so far and may represent targets for future research. Examples include security and privacy aspects, as well as verification and validation in connection with TOSCA models. Hence we expect TOSCA to remain a relevant topic for future cloud research.

Acknowledgment. This work received funding from the European Union's Horizon 2020 research and innovation programme (grant agreement 731678 (RestAssured)).

References

1. Alonso, J., Orue-Echevarria, L., Escalante, M.: Cloud compliant applications: A reference framework to assess the maturity of software applications with respect to cloud. In: Proc. MESOCA 2015. pp. 41–45 (2015)
2. Antonenko, V., Smeliansky, R., Ermilov, A., Plakunov, A., Pinaeva, N., Romanov, A.: C2: General purpose cloud platform with nfv life-cycle management. In: Proc. CloudCom 2017. vol. 2017-December, pp. 353–356 (2017)
3. Antonenko, V., Smeliansky, R., Ermilov, A., Romanov, A., Pinaeva, N., Plakunov, A.: Cloud infrastructure for researchers basing on nfvmanagement and orchestration. In: Proc. NEC 2017. vol. 2023, pp. 75–81 (2017)
4. Antoniades, D., Loulloudes, N., Foudoulis, A., Sophokleous, C., Trihinas, D., Pallis, G., Dikaiakos, M., Kornmayer, H.: Enabling Cloud Application Portability. In: Proc. UCC 2015. pp. 354–360 (2015)
5. Binz, T., Breitenbücher, U., Haupt, F., Kopp, O., Leymann, F., Nowak, A., Wagner, S.: OpenTOSCA – A runtime for TOSCA-based cloud applications. In: Proc. ICSOC 2013. pp. 692–695 (2013)

6. Binz, T., Breitenbücher, U., Kopp, O., Leymann, F.: TOSCA: Portable Automated Deployment and Management of Cloud Applications. In: Advanced Web Services. pp. 527–549 (2014)
7. Binz, T., Breitenbücher, U., Kopp, O., Leymann, F., Weiß, A.: Improve resource-sharing through functionality-preserving merge of cloud application topologies. In: Proc. CLOSER 2013. pp. 96–103 (2013)
8. Binz, T., Breiter, G., Leyman, F., Spatzier, T.: Portable Cloud Services Using TOSCA. IEEE Internet Computing **16**(3), 80–85 (2012)
9. Bonchi, F., Brogi, A., Canciani, A., Soldani, J.: Behaviour-Aware Matching of Cloud Applications. In: Proc. TASE 2016. pp. 117–124 (2016)
10. Breitenbücher, U., Binz, T., Képes, K., Kopp, O., Leymann, F., Wettinger, J.: Combining declarative and imperative cloud application provisioning based on TOSCA. In: Proc. IC2E 2014. pp. 87–96 (2014)
11. Breitenbücher, U., Binz, T., Kopp, O., Képes, K., Leymann, F., Wettinger, J.: Hybrid TOSCA provisioning plans: Integrating declarative and imperative cloud application provisioning technologies. In: Proc. CLOSER 2015. pp. 239–262 (2016)
12. Breitenbücher, U., Binz, T., Kopp, O., Leymann, F.: Vinothek – a self-service portal for TOSCA. In: Proc. ZEUS 2014. pp. 72–75 (2014)
13. Breitenbücher, U., Binz, T., Kopp, O., Leymann, F., Schumm, D.: Vino4TOSCA: A visual notation for application topologies based on TOSCA. In: Proc. OTM 2012. pp. 416–424 (2012)
14. Breitenbücher, U., Kopp, O., Leymann, F., Wettinger, J.: Towards integrating TOSCA and ITIL. In: Proc. ZEUS 2013. vol. 1029, pp. 28–31 (2013)
15. Breiter, G., Behrendt, M., Gupta, M., Moser, S., Schulze, R., Sippli, I., Spatzier, T.: Software defined environments based on TOSCA in IBM cloud implementations. IBM Journal of Research and Development **58**(2) (2014)
16. Brereton, P., Kitchenham, B.A., Budgen, D., Turner, M., Khalil, M.: Lessons from applying the systematic literature review process within the software engineering domain. Journal of Systems and Software **80**(4), 571–583 (2007)
17. Brogi, A., Canciani, A., Soldani, J.: Modelling and analysing cloud application management. In: Proc. ESOCC 2015. pp. 19–33 (2015)
18. Brogi, A., Canciani, A., Soldani, J., Wang, P.: Modelling the behaviour of management operations in cloud-based applications. In: Proc. PNSE 2015. pp. 191–205 (2015)
19. Brogi, A., Canciani, A., Soldani, J., Wang, P.: A Petri Net-Based Approach to Model and Analyze the Management of Cloud Applications. Proc. ToPNoC 2015 **11**, 28–48 (2016)
20. Brogi, A., Carrasco, J., Cubo, J., Di Nitto, E., Durán, F., Fazzolari, M., Ibrahim, A., Pimentel, E., Soldani, J., Wang, P., D'Andria, F.: Adaptive management of applications across multiple clouds: The seaclouds approach. CLEI Electron. J. **18**(1) (2015)
21. Brogi, A., Cifariello, P., Soldani, J.: DrACO: Discovering available cloud offerings. Computer Science - Research and Development **32**(3-4), 269–279 (2017)
22. Brogi, A., Soldani, J.: Matching cloud services with TOSCA. In: Proc. ESOCC 2013. pp. 218–232 (2013)
23. Brogi, A., Soldani, J.: Reusing cloud-based services with TOSCA. In: Proc. Informatik 2014. pp. 235–246 (2014)
24. Brogi, A., Soldani, J.: Finding available services in TOSCA-compliant clouds. Science of Computer Programming **115-116**, 177–198 (2016)
25. Brogi, A., Soldani, J., Wang, P.: TOSCA in a nutshell: Promises and perspectives. In: Proc. ESOCC 2014. pp. 171–186 (2014)

26. Calcaterra, D., Cartelli, V., Di Modica, G., Tomarchio, O.: Combining TOSCA and BPMN to enable automated cloud service provisioning. In: Proc. CLOSER 2017. pp. 159–168 (2017)
27. Cardoso, J., Binz, T., Breitenbücher, U., Kopp, O., Leymann, F.: Cloud computing automation: Integrating USDL and TOSCA. In: Proc. CAiSE 2013. pp. 1–16 (2013)
28. Carrasco, J., Cubo, J., Durn, F., Pimentel, E.: Bidimensional cross-cloud management with TOSCA and Brooklyn. In: Proc. CLOUD 2016. pp. 951–955 (2017)
29. Carrasco, J., Cubo, J., Pimentel, E.: Towards a flexible deployment of multi-cloud applications based on tosca and camp. Communications in Computer and Information Science **508**, 278–286 (2015)
30. Chappell, C.: Deploying Virtual Network Functions: The Complementary Roles of TOSCA & NETCONF/YANG. In: White Paper (2015)
31. Chareonsuk, W., Vatanawood, W.: Formal verification of cloud orchestration design with TOSCA and BPEL. In: Proc. ECTI-CON 2016. pp. 1–5 (2016)
32. Da Silva, A., Breitenbücher, U., Hirmer, P., Képes, K., Kopp, O., Leymann, F., Mitschang, B., Steinke, R.: Internet of things out of the box: Using TOSCA for automating the deployment of IoT environments. In: Proc. CLOSER 2017. pp. 330–339 (2017)
33. Di Martino, B., Cretella, G., Esposito, A.: Defining cloud services workflow: a comparison between TOSCA and OpenStack Hot. In: Proc. CISIS 2015. pp. 541–546 (2015)
34. Di Martino, B., Cretella, G., Esposito, A.: Research Initiatives and Emerging Standards, pp. 93–121. Springer International Publishing, Cham (2015)
35. Di Martino, B., Cretella, G., Esposito, A.: A comparison between TOSCA and OpenStack HOT through cloud patterns composition. International Journal of Grid and Utility Computing **8**(4), 299–311 (2017)
36. Dräxler, S., Karl, H., Mann, Z.A.: Joint optimization of scaling and placement of virtual network services. In: Proc. CCGrid 2017. pp. 365–370 (2017)
37. Endres, C., Breitenbücher, U., Leymann, F., Wettinger, J.: Anything to topology – a method and system architecture to topologize technology-specific application deployment artifacts. In: Proc. CLOSER 2017. pp. 180–190 (2017)
38. Glaser, F.: Domain model optimized deployment and execution of cloud applications with TOSCA. Proc. SAM 2016 **9959 LNCS**, 68–83 (2016)
39. Glaser, F., Erbel, J., Grabowski, J.: Model driven cloud orchestration by combining TOSCA & OCCI. In: Proc. CLOSER 2017. pp. 644–650 (2017)
40. Gusev, M., Kostoska, M., Ristov, S.: Cloud P-TOSCA porting of N-Tier applications. In: Proc. TELFOR 2014. pp. 935–938 (2014)
41. Han, R., Ghanem, M., Guo, Y.: Elastic-tosca: supporting elasticity of cloud application in TOSCA. In: CLOUD COMPUTING 2013. pp. 93–100 (2013)
42. Harrer, S., Lenhard, J., Wirtz, G., Van Lessen, T.: Towards uniform BPEL engine management in the cloud. In: Proc. INFORMATICS 2014. vol. P-232, pp. 259–270 (2014)
43. Haupt, F., Leymann, F., Nowak, A., Wagner, S.: Lego4TOSCA: Composable building blocks for cloud applications. In: Proc. CLOUD 2014. pp. 160–167 (2014)
44. Healy, P., Meyer, S., Morrison, J., Lynn, T., Paya, A., Marinescu, D.: Bid-centric cloud service provisioning. In: Proc. ISPDC 2014. pp. 73–81 (2014)
45. Hirmer, P., Breitenbücher, U., Binz, T., Leymann, F.: Automatic topology completion of TOSCA-based cloud applications. In: Proc. Informatik 2014. pp. 247–258 (2014)
46. Hung, Y.M., Chien, S.C., Chunghwa, Y.Y.: Orchestration of NFV virtual applications based on TOSCA data models. In: Proc. APNOMS 2017. pp. 219–222 (2017)

47. Ivanovska, B., Ristov, S., Kostoska, M., Gusev, M.: Using the P-TOSCA model for energy efficient cloud. In: Proc. MIPRO 2015. pp. 245–249 (2015)
48. Katsaros, G., Menzel, M., Lenk, A., Rake-Revelant, J., Skipp, R., Eberhardt, J.: Cloud application portability with TOSCA, Chef and Openstack: Experiences from a proof-of-concept implementation. In: Proc. IC2E 2014. pp. 295–302 (2014)
49. Képes, K., Breitenbücher, U., Leymann, F.: The SePaDe system: Packaging entire XaaS layers for automatically deploying & managing applications. In: Proc. CLOSER 2017. pp. 626–635 (2017)
50. Komarek, A., Pavlik, J., Sobeslav, V.: Hybrid system orchestration with TOSCA and salt. Journal of Engineering and Applied Sciences **12**(9), 2396–2401 (2017)
51. Kopp, O., Binz, T., Breitenbücher, U., Leymann, F.: BPMN4TOSCA: A domain-specific language to model management plans for composite applications. In: Proc. BPMN 2012. pp. 38–52 (2012)
52. Kopp, O., Binz, T., Breitenbücher, U., Leymann, F.: Winery – a modeling tool for TOSCA-based cloud applications. In: Proc. ICSOC 2013. pp. 700–704 (2013)
53. Kopp, O., Binz, T., Breitenbücher, U., Leymann, F., Michelbach, T.: A domain-specific modeling tool to model management plans for composite applications. In: Proc. ZEUS 2015. pp. 51–54 (2015)
54. Kostoska, M., Chorbev, I., Gusev, M.: Creating portable TOSCA archive for iKnow university management system. In: Proc. FedCSIS 2014. pp. 761–768 (2014)
55. Kostoska, M., Donevski, A., Gusev, M., Ristov, S.: Porting an N-tier application on cloud using P-TOSCA: A case study. In: Proc. MIPRO 2015. pp. 281–285 (2015)
56. Kostoska, M., Gusev, M., Ristov, S.: A new cloud services portability platform. In: Proc. DAAAM 2014. vol. 69, pp. 1268–1275 (2014)
57. Li, F., Vögler, M., Claeßens, M., Dustdar, S.: Towards automated IoT application deployment by a cloud-based approach. In: Proc. SOCA 2013. pp. 61–68 (2013)
58. Lipton, P.: Escaping vendor lock-in with tosca, an emerging cloud standard for portability. In: CA Technology Exchange 4. pp. 49–55 (2013)
59. Mann, Z.Á.: Resource optimization across the cloud stack. IEEE Transactions on Parallel and Distributed Systems **29**(1), 169–182 (2018)
60. Mann, Z.Á., Metzger, A.: Optimized cloud deployment of multi-tenant software considering data protection concerns. In: Proc. CCGrid 2017. pp. 609–618 (2017)
61. Molto, G., Caballer, M., Perez, A., Alfonso, C., Blanquer, I.: Coherent application delivery on hybrid distributed computing infrastructures of virtual machines and Docker containers. In: Proc. PDP 2017. pp. 486–490 (2017)
62. Neubauer, P., Bergmayr, A., Mayerhofer, T., Troya, J., Wimmer, M.: XMLText: From XML schema to Xtext. In: Proc. SLE 2015. pp. 71–76 (2015)
63. Nowak, A., Binz, T., Fehling, C., Kopp, O., Leymann, F., Wagner, S.: Pattern-driven green adaptation of process-based applications and their runtime infrastructure. Computing **94**(6), 463–487 (2012)
64. OASIS: Web Services Business Process Execution Language Version 2.0 (Apr 2007), OASIS Standard
65. OASIS: Topology and Orchestration Specification for Cloud Applications Version 1.0. (Nov 2013), http://docs.oasis-open.org/tosca/TOSCA/v1.0/os/TOSCA-v1.0-os.html., OASIS Standard
66. OMG: Business Process Model and Notation (BPMN) Version 2.0. (Jan 2011), OMG Document Number: formal/2011-01-03
67. Palma, D., Rutkowski, M., Spatzier, T.: TOSCA Simple Profile in YAML Version 1.0 (Dec 2016), http://docs.oasis-open.org/tosca/TOSCA-Simple-Profile-YAML/v1.0/TOSCA-Simple-Profile-YAML-v1.0.html., OASIS Standard

68. Panarello, A., Breitenbücher, U., Leymann, F., Puliafito, A., Zimmermann, M.: Automating the deployment of multi-cloud applications in federated cloud environments. In: Proc. ValueTools 2016. pp. 194–201 (2017)
69. Qanbari, S., Li, F., Dustdar, S.: Toward portable cloud manufacturing services. IEEE Internet Computing **18**(6), 77–80 (2014)
70. Qanbari, S., Sebto, V., Dustdar, S.: Cloud resources-events-agents model: Towards TOSCA-based applications. Proc. ESOCC 2014 **8745 LNCS**, 160–170 (2014)
71. Qanbari, S., Zadeh, S., Vedaei, S., Dustdar, S.: CloudMan: A platform for portable cloud manufacturing services. In: Proc. IEEE Big Data 2014. pp. 1006–1014 (2015)
72. Qasha, R., Cala, J., Watson, P.: Towards Automated Workflow Deployment in the Cloud Using TOSCA. In: Proc. CLOUD 2015. pp. 1037–1040 (2015)
73. Qasha, R., Cala, J., Watson, P.: Dynamic deployment of scientific workflows in the cloud using container virtualization. In: Proc. CloudCom 2016. pp. 269–276 (2017)
74. Razaq, A., Tianfield, H., Barrie, P., Yue, H.: Service broker based on cloud service description language. In: Proc. ISPDC 2016. pp. 196–201 (2017)
75. Saatkamp, K., Breitenbcher, U., Leymann, F., Wurster, M.: Generic driver injection for automated IoT application deployments. In: Proc. iiWAS2017. pp. 320–329 (2017)
76. Saatkamp, K., Breitenbücher, U., Kopp, O., Leymann, F.: Topology splitting and matching for multi-cloud deployments. In: Proc. CLOSER 2017. pp. 247–258 (2017)
77. Salsano, S., Lombardo, F., Pisa, C., Greto, P., Blefari-Melazzi, N.: Rdcl 3d, a model agnostic web framework for the design and composition of nfv services. In: Proc. NFV-SDN 2017. vol. 2017-January, pp. 216–222 (2017)
78. Sampaio, A., Rolim, T., Mendona, N., Cunha, M.: An approach for evaluating cloud application topologies based on TOSCA. In: Proc. CLOUD 2016. pp. 407–414 (2017)
79. Franco da Silva, A., Hirmer, P., Breitenbücher, U., Kopp, O., Mitschang, B.: Customization and provisioning of complex event processing using TOSCA. Computer Science – Research and Development pp. 1–11 (2017)
80. Silva, G., Rose, L., Calinescu, R.: Cloud dsl: A language for supporting cloud portability by describing cloud entities. In: CEUR Workshop Proceedings. vol. 1242, pp. 36–45 (2014)
81. Soldani, J., Binz, T., Breitenbücher, U., Leymann, F., Brogi, A.: ToscaMart: A method for adapting and reusing cloud applications. Journal of Systems and Software **113**, 395–406 (2016)
82. Steimle, F., Wieland, M., Mitschang, B., Wagner, S., Leymann, F.: Extended provisioning, security and analysis techniques for the ECHO health data management system. Computing **99**(2), 183–201 (2017)
83. Sundararajan, P., Durairajan, S.: Portable service management deployment over cloud platforms to support production workloads. In: Proc. CCEM 2013 (2013)
84. Sungur, C., Kopp, O., Leymann, F.: Supporting informal processes. In: CEUR Workshop Proceedings. vol. 1140, pp. 49–56 (2014)
85. Tricomi, G., Panarello, A., Merlino, G., Longo, F., Bruneo, D., Puliafito, A.: Orchestrated multi-cloud application deployment in OpenStack with TOSCA. In: Proc. SMARTCOMP 2017. pp. 1–6 (2017)
86. Tsigkanos, C., Kehrer, T.: On formalizing and identifying patterns in cloud workload specifications. In: Proc. WICSA 2016. pp. 262–267 (2016)
87. Ulbricht, S., Amme, W., Heinze, T., Moser, S., Wehle, H.D.: Portable green cloud services. In: Proc. CLOSER 2014. pp. 53–59 (2014)
88. Vetter, A.: Detecting operator errors in cloud maintenance operations. In: Proc. CloudCom 2016. pp. 639–644 (2017)

89. Waizenegger, T., Wieland, M., Binz, T., Breitenbücher, U., Haupt, F., Kopp, O., Leymann, F., Mitschang, B., Nowak, A., Wagner, S.: Policy4TOSCA: A policy-aware cloud service provisioning approach to enable secure cloud computing. In: Proc. OTM 2013. pp. 360–376 (2013)
90. Waizenegger, T., Wieland, M., Binz, T., Breitenbücher, U., Leymann, F.: Towards a policy-framework for the deployment and management of cloud services. In: Proc. SECURWARE 2013. pp. 14–18 (2013)
91. Wettinger, J., Behrendt, M., Binz, T., Breitenbücher, U., Breiter, G., Leymann, F., Moser, S., Schwertle, I., Spatzier, T.: Integrating configuration management with model-driven cloud management based on TOSCA. In: Proc. CLOSER 2013. pp. 437–446 (2013)
92. Wettinger, J., Binz, T., Breitenbücher, U., Kopp, O., Leymann, F.: Streamlining cloud management automation by unifying the invocation of scripts and services based on TOSCA. In: Cloud Technology: Concepts, Methodologies, Tools, and Applications, vol. 4, pp. 2240–2261. IGI Global (2014)
93. Wettinger, J., Binz, T., Breitenbücher, U., Kopp, O., Leymann, F., Zimmermann, M.: Unified invocation of scripts and services for provisioning, deployment and management of cloud applications based on TOSCA. In: Proc. CLOSER 2014. pp. 559–568 (2014)
94. Wettinger, J., Breitenbücher, U., Kopp, O., Leymann, F.: Streamlining DevOps automation for Cloud applications using TOSCA as standardized metamodel. Future Generation Computer Systems **56**, 317–332 (2016)
95. Wettinger, J., Breitenbücher, U., Leymann, F.: Standards-based DevOps automation and integration using TOSCA. In: Proc. UCC 2014. pp. 59–68 (2014)
96. Wurster, M., Breitenbcher, U., Falkenthal, M., Leymann, F.: Developing, deploying, and operating twelve-factor applications with tosca. In: Proc. iiWAS2017. pp. 519–525 (2017)
97. Yongsiriwit, K., Sellami, M., Gaaloul, W.: A semantic framework supporting cloud resource descriptions interoperability. In: Proc. CLOUD 2016. pp. 585–592 (2017)
98. Yoshida, H., Ogata, K., Futatsugi, K.: Formalization and verification of declarative cloud orchestration. In: Proc. ICFEM 2015. pp. 33–49 (2015)
99. Zimmermann, M., Breitenbücher, U., Leymann, F.: A TOSCA-based programming model for interacting components of automatically deployed cloud and IoT applications. In: Proc. ICEIS 2017. vol. 2, pp. 121–131 (2017)