Institute of Architecture of Application Systems

Towards Deployable Research Object Archives Based on TOSCA

Michael Zimmermann[1], Uwe Breitenbücher[1], Jasmin Guth[1],
Sibylle Hermann[2], Frank Leymann[1], and Karoline Saatkamp[1]

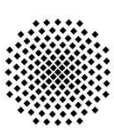
[1]Institute of Architecture of Application Systems, University of Stuttgart, Germany
[firstname.lastname]@iaas.uni-stuttgart.de

[2]University Library of Stuttgart, Germany
sibylle.hermann@ub.uni-stuttgart.de

BibTeX:

```
@inproceedings{Zimmermann2018_ResearchObjectArchives,
  author    = {Zimmermann, Michael and Breitenb\"{u}cher, Uwe and Guth, Jasmin
               and Hermann, Sibylle and Leymann, Frank and Saatkamp, Karoline},
  title     = {{Towards Deployable Research Object Archives Based on TOSCA}},
  booktitle = {Papers from the 12\textsuperscript{th} Advanced Summer School on
               Service-Oriented Computing (SummerSoC 2018)},
  year      = {2018},
  pages     = {31--42},
  publisher = {IBM Research Division}
}
```

The full version of this publication has been presented as a poster at the Advanced Summer School on Service Oriented Computing (SummerSOC 2018).
http://www.summersoc.eu



University of Stuttgart
Germany

Towards Deployable Research Object Archives Based on TOSCA

Michael Zimmermann[1], Uwe Breitenbücher[1], Jasmin Guth[1], Sibylle Hermann[2], Frank Leymann[1], and Karoline Saatkamp[1]

[1] Institute of Architecture of Application Systems, University of Stuttgart
Universitätsstraße 38, 70569 Stuttgart, Germany
{firstname.lastname}@iaas.uni-stuttgart.de
[2] University Library of Stuttgart
Holzgartenstraße 16, 70174 Stuttgart, Germany
sibylle.hermann@ub.uni-stuttgart.de

Abstract. In science, reproducibility means that a scientific experiment can be repeated by another scientist with the same result. This is of particular importance to verify the results as well as to show the usefulness and reusability for further research. However, the exclusive publication of the research results in a scientific journal is usually not sufficient. In addition to research results, also research data as well as research software need to be published and made public available in order to enable researcher to gain new insights and thus advance research. However, the reproducibility and reusability of research data and research software typically is hindered by several barriers. Therefore, this work intends to first provide an overview of the current situation and issues in this particular topic and furthermore sketch our vision of standards-based Research Object Archives containing scientific publications, software, data, metadata and licenses in order to tackle the existing problems.

Keywords: Research Object, Reusability, Reproducibility, Deployment Model, TOSCA.

1 Introduction and Background

New findings in all research areas are based on the fact that existing knowledge and results of scientific studies and experiments can be shared, verified, and reused. This is of particular importance to verify the results, and thus verify the statements of the experiment as well as to show the usefulness and reusability for further research. Typically, only the scientific findings are published in scientific journals. The rapidly growing collective knowledge is dependent on software processing the raw data, which is an essential part of all scientific work today. Therefore, in order to understand and reproduce research results, not only the publication of the results and the research data, but also the associated research software must be made publicly available [2,28]. However, research software does not only have to be published, but must also be

easily executable in order to enable researcher to reuse the software or reproduce the research results. But the heterogeneity in used infrastructures and technologies as well as other different technical requirements for the provisioning and operation of scientific applications requires expert IT knowledge about, for example, deployment technologies in order to be able to use the developed research software. Thus, some standards-based approach to automate the provisioning, the management, as well as the execution of research software and scientific experiments is required.

Regarding only the pure management of research software and research data, there is already some work available. That not only the research results and publications, but also related research software and research data is required in order to make research software usable and research results comprehensible has already been recognized and discussed by different authors [2,4,12,28]. Moreover, in various research domains, there are already some first isolated solutions, which are mainly concerned with the provisioning of research software and the connection with the generated data, for example, in the area of high-performance computing [26] or neuroscience [20]. There are also first approaches to model and store research results together with algorithms, data, methods, workflows and metadata [3–5,15]. Hunter [15] proposes so-called *Scientific Publication Packages*, which focus on the description of artifacts and the relationships between them. Bechhofer et al. [3,4] and Belhajjame et al. [5] define so-called *Research Objects*. Both approaches focus primarily on linking data with associated artifacts, such as algorithms or used methods, as well as the representation of the experiment workflow. Research Objects also have a defined life-cycle and are versionable. This way, different states, such as the progress of a study, can be directly mapped within the research object and changes to the object can be traced. However, both approaches do not address the self-contained packaging and automated provision of research software locally or in a cloud environment, the citability of research software and data, or the integration of license checks of the source code. With the constantly increasing importance, not only of research data and research software, but also the linking with the publications, new challenges for researcher arise: There is uncertainty regarding the licensing of software, there are no clear guidelines on how research software can be described and thus, made discoverable by metadata, and there are no procedures for the automated installation of research software to make it usable [1]. In addition, mechanisms are needed to ensure the sustainable storage and retrievability, the identifiability and thus the citability of different versions of research data, software or required components. Even though, the concepts of Scientific Publication Packages and Research Objects show the first approaches for dealing with research results, research data, and research software, a comprehensive and standards-based solution not only for storage, but also for finding, licensing, citation and provisioning of any research software has not yet been developed. Moreover, the aspects of automatically provisioning as well as managing applications is missing in these concepts. Thus, we want to use the findings from these concepts and further develop a standards-based packaging format enabling the automated provisioning and management of applications for the requirements of research software.

Regarding the automated deployment of cloud applications, in recent years, several technologies and standards were developed. This includes configuration management technologies such as Chef[1], Ansible[2], or Puppet[3], as well as container technologies such as Docker[4]. However, configuration management technologies and container technologies are based on specific artifacts, for example, installation scripts required to deploy the application. Furthermore, these artifacts and their formats differ greatly dependent of the used technologies [11]. Therefore, when multiple technologies need to be combined to deploy non-trivial applications, broad technical knowledge of the technologies and an integration process are required. But, as mentioned before, since information technology is being used in more and more scientific areas and not only in computer science, this technical knowledge cannot be assumed in order to run research software. Besides these technologies, there are also standards such as the Topology and Orchestration Specification for Cloud Applications (TOSCA) [8,22,23]. TOSCA is an OASIS standard that enables the definition of application deployments by topology models and management plans, which can be executed automatically by a TOSCA runtime, like e.g., the OpenTOSCA container [7]. A topology model describes the application components and their relations to each other. This includes application-specific components, such as PHP applications or databases, middleware, and infrastructure components, such as web servers or virtual machines. Thereby, application deployments can be described in a vendor-independent and portable manner. Thus, we want to use the TOSCA concepts in order to realize the provisioning and managing aspect of our Research Object Archives approach.

To sum up, there is no comprehensive approach available yet enabling the packaging of all required research artifacts as well as the corresponding publication, that is also supporting the automated provisioning and managing of the research software. Therefore, in this paper, we want to introduce our vision of a standards-based approach for packaging all scientific artifacts, such as research results, research data, research software as well as scientific publications together into one executable archive, thus, enabling the automated provisioning and management of scientific applications. To achieve that, we try to combine the packaging concepts of Research Objects with the deployment and managing concepts of the TOSCA standard in order to create automatically deployable Research Object Archives.

The remainder of this paper is structured as follows: We discuss and present the fundamentals, such as Research Objects and the OASIS standard Topology Orchestration and Specification for Cloud Applications (TOSCA) in detail in Section 2. We sketch our idea of Research Object Archives in Sect. 3 and discuss how it addresses the identified issues and how it enables the portability by an approach enabling the automated deployment and managing. In Sect. 4, works related to our approach are discussed. Finally, Sect. 5 concludes this paper.

---

1 https://www.chef.io/chef
2 https://www.ansible.com
3 https://puppet.com
4 https://www.docker.com

2 Basic Concepts

The idea behind Research Object Archives (ROARs) is to give the possibility to publish automatically deployable research software. For this purpose, the concept of Research Objects (ROs) will be implemented with the Topology Orchestration and Specification for Cloud Applications (TOSCA) standard. This section will introduce the basic concepts of ROs and TOSCA.

2.1 Research Objects

In general, reuse of a specific set of research results requires additional information. To meet this requirement, Bechhhofer et al. developed the concept of Research Objects. [3,4] They define the following set of principles to publish research results together with algorithms, data, methods, workflows, and metadata:

Reusable: ROs should be usable as a whole or in parts.

Repurposeable: The relationship of the contained parts should be described.

Repeatable: There should be enough information to repeat the research. Also important are sufficient privileges to access the data.

Reproducible: There should be enough information to validate the result.

Replayable: There should be enough information to understand what happens in a specific process.

Referenceable: A RO need a unique Identifier to get credit over citations for the research output.

To implement these principles, Bechhofer et al. propose the following features for ROs. References and resources are *Aggregated* in an RO. Such aggregation has to organize the content to resolve the resources dynamically. The ROs as object and the content of the RO have a separate *Identity*. Thus, the entire research object or parts of it can be unambiguously referenced. *Metadata* has two functions: To discover the RO and to describe the research for reuse. The RO as a research result must be described by machine-readable metadata to be found. Additionally, the content requires information about licensing, attribution, copyright or descriptions of provenance and the derivation of results. As research is a process with a *Lifecycle* that should be described. Different events will happen in a particular sequence and different actions can be performed at various stages. These changes over time, particularly adjustments like deleting or adding content to a RO, must be recorded with *Versioning*. The *Management* of ROs needs the possibility to operations like Creation, Retrieval, Update, and Deletion (CRUD). *Security* issues like access, authentication, ownership, and trust need to be addressed. The last feature Bechhofer et al. propose is the *Graceful Degradation of Understanding* to give the opportunity to use the Research object without understanding the whole research process.

The approach of Research Objects do not address a specific implementation of ROs. Neither the self-descriptive packaging and automated provision of research software locally or in a cloud nor the question of how to cite the research software and data or the integration of license checks of the source code is addressed by Research Objects.

2.2 Topology Orchestration and Specification for Cloud Applications

Since our approach is based on the TOSCA standard, we introduce the OASIS standard Topology Orchestration and Specification for Cloud Applications (TOSCA) in this subsection, in order to provide a comprehensive background. TOSCA enables to describe the automated deployment and management of applications in an interoperable and portable manner. We only give an overview of the fundamental TOSCA concepts, detailed information can be found in the TOSCA Specifications [23,24], the TOSCA Primer [24] and in Binz et al. [8]. A comparison of TOSCA with other Cloud Modeling languages can be found in Bergmayr et al. [6].

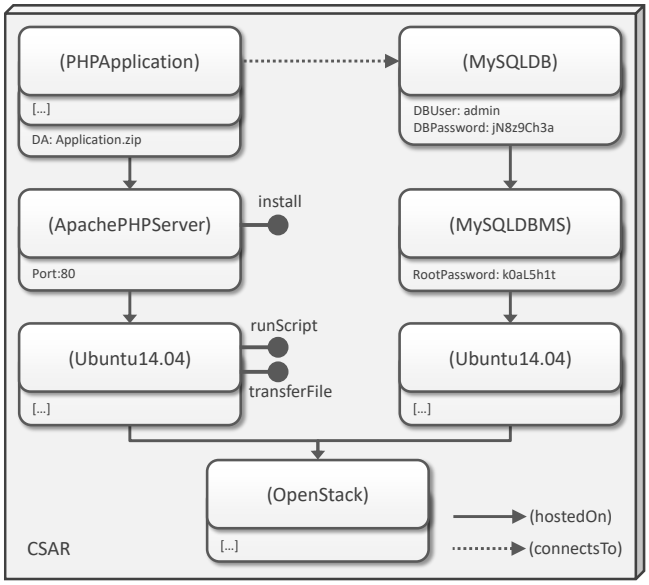


Fig. 1. Exemplary TOSCA Topology Template.

The TOSCA standards enables to describe required infrastructure resources, software components, as well as the structure of cloud or IoT applications. Furthermore, TOSCA enables to define the required operations for managing such applications. Thus, TOSCA enables the automated deployment and management of cloud and IoT applications. The structure of cloud applications are defined by so-called TOSCA *Topology Templates*. An exemplary TOSCA Topology Template, following the visual notation Vino4TOSCA [10] is illustrated in Fig. 1. Technically, a Topology Template is a graph consisting of nodes and directed edges. The nodes represent components of the application, for instance, a virtual machine or a database and are called *Node Templates*. The edges connecting the nodes specify the relations between Node Templates, for example, “hostedOn” or “connectsTo” and are called *Relationship Templates*. For reusability purposes, the TOSCA standard enables the specification of *Node Types* and *Relationship Types* defining the semantics of the Node Templates and Relationship Templates. Node

Types, for example, enable to define *Properties* as well as *Management Operations*. Properties are for example passwords, usernames, or the port of a web server and thus, enable the customization of the TOSCA Topology Templates. Management operations enable the management of the modeled components. For example, typically a software component node provides an "install" operation in order to install the component or a hypervisor node provides a "createVM" and "terminateVM" operation in order to create and terminate virtual machines.

Management operations are implemented by so-called *Implementation Artifacts (IAs)*. IAs can be implemented using any various technologies, e.g., as a WAR-file providing a WSDL-based SOAP Web Service, a configuration management technology, such as Ansible, Puppet, Chef, or just as a simple shell script. Besides IAs, TOSCA also defines so-called *Deployment Artifacts (DAs)* representing the artifacts implementing the business logic of the components of an application. For example, the DA of a Java application can be a WAR-file. In case of a PHP application, a DA would be a ZIP file containing all PHP files, images, and other required files. The creation and termination of instances of a modeled application can either be done *declaratively*, by deriving the actions that need to be executed directly from the Topology Template or *imperatively* with help of *Management Plans* [14]. A Management Plan is an executable workflow model specifying all tasks as well as the order in which these tasks need to be executed to achieve a certain management functionality, e.g., the provisioning of a new application instance or to scale out a component of a running application instance. TOSCA does not specify how such plans should be implemented, however, recommends using a workflow language such as the *Business Process Execution Language (BPEL)* [21] or the *Business Process Model and Notation (BPMN)* [25]. Furthermore, there is a BPMN extension called *BPMN4TOSCA* that is explicitly tailored for describing TOSCA-based management plans [17,19].

Additionally, the TOSCA standard also specifies a portable and self-contained packaging format, so-called *Cloud Service Archive (CSAR)*. CSARs enable to package Topology Templates, type definitions, Management Plans, IAs, DAs, and all other required files for automating the provisioning as well as the management of applications into one archive. Through the standardized meta-model and packaging format, these CSARs can be automatically processed and executed by standard-compliant TOSCA Runtime Environments, such as the OpenTOSCA Ecosystem[5]. Therefore, portability as well as interoperability can be ensured.

Overall, the OASIS standard TOSCA enables to automate the provisioning and management of complex applications as well as to specify and package all descriptions and required files in a portable format. Therefore, the standard provides a suitable basis for packaging and managing research software together with associated artifacts for an automated provisioning. Up to now, the identification via metadata as well as the linking with licenses, external data sources, and publication repositories is not considered in the standard. However, due to the extensibility of the specification, this standard can be extended accordingly.

[5] https://github.com/OpenTOSCA

3 Towards an Approach for the Automated Deployment of Research Objects

In this section, we introduce our approach of portable Research Object Archives (ROARs). Therefore, we first present the main concepts of ROARs and subsequently illustrate how they are supposed to be created and used. We further show, how they enable the automated deployment of the modeled application and how the proposed features presented in Sect. 2.1 are realized by ROARs.

3.1 Research Object Archive

In order to enable developing, packaging, publishing, and deploying research software efficiently, a self-contained and portable packaging format for research software is inevitable. Thus, the main goal objectives are to develop a packaging format that enables bundling all important information of research software, especially its technical dependencies, associated research data, descriptive metadata, used licenses, and a reference to the corresponding publication, as well as to enable the automated provisioning. The conceptual structure of a ROAR is depicted in Fig. 2. The ROAR format is based on Research Objects (ROs) (cf. Sect. 2.1) and the provisioning and management concepts of the TOSCA standard (cf. Sect. 2.2), which is explained in more detail in the following.

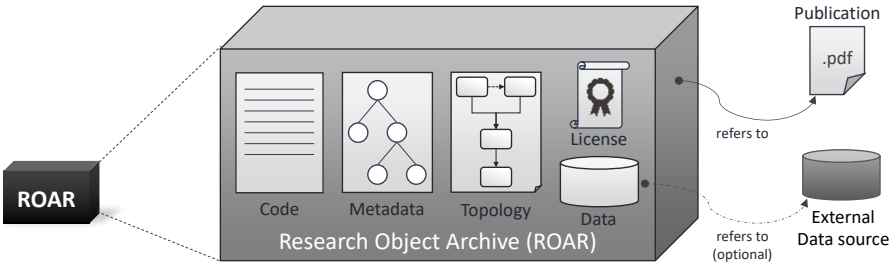


Fig. 2. Structure of a Research Object Archive (ROAR).

For ensuring the linking to the corresponding publication, a ROAR, for example in the case of an open access publication, should refer directly to the publication – typically a PDF file – and in case of house publications to the landing page of the article at the respective publisher. Furthermore, a ROAR should contain the used source code, descriptive metadata, license information, and the used database, which can also be located in an external repository. By supporting metadata, the ROAR can be found easily and thus, supporting the proposed feature *Metadata* in Sect. 2.1. Since database can be located in external repositories, ROARs need to support referencing external data sources and resolving them dynamically (cf. *Aggregation* in Sect. 2.1). Moreover, because data

or software can change, ROARs need to support versioning (cf. *Versioning* in Sect. 2.1). Based on the metadata, a unique ID can be assigned to the ROAR, thus enabling citation (cf. *Identity* in Sect. 2.1). Of course, ROARs should be created, updated, and retrieved by researcher as well as stored in repositories or archives, thus supporting the proposed feature *Management* in Sect. 2.1. Furthermore, by not only adding the research software, dependencies, as well as required data, but also a TOSCA-based description how the application can be provisioned, a ROAR should enable the automated provisioning of the modeled and archived application and thus, simplify the repeatability of the research results (cf. *Graceful Degradation of Understanding* in Sect. 2.1). However, since not all necessary functionalities, such as referencing publications in a CSAR, are currently supported by the TOSCA standard, extensions of TOSCA and the CSAR packaging format must be developed and combined with the RO concept.

3.2 Research Object Portability and Deployment Approach

The reusability as well as the reproducibility are vital factors in science. Thus, ROARs need to be designed in a way supporting (i) the researcher creating such a ROAR as well as (ii) the researcher that wants to reuse a modeled application. In this section, we illustrate how ROARs are created and how they can be reused.

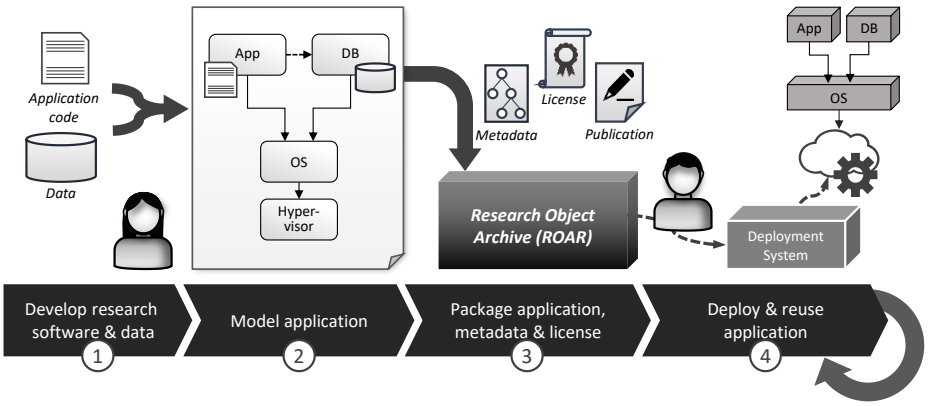


Fig. 3. Overview: Usage of Deployable Research Object Archives.

Fig. 3 shows the conceptual approach, the roles involved, and the general procedure for storing, testing, finding, and reusing research software as well as the associated data. When scientists want to publish their developed research software with associated data (see step 1), they first have to model the application's topology with all its dependencies as well as required data (see step 2) using the TOSCA modeling concepts (cf. Sect. 2.2). In our approach, it should also be possible to insert the data directly to the final ROAR as well as to only insert a reference to an external location where

the data is stored. Thus, the topology model has to support both variants to define required data. Since, our concept is based on the TOSCA standard, of course available modeling tools supporting the TOSCA standard can be used for modeling the application, for example, the open-source tool *Winery*[6] [18]. After the modeling, the research software can be packaged with associated data, licenses, descriptive metadata, as well as associated publications, published, and persistently stored as a ROAR (see step 3). The selection of a suitable license for the research software can be supported by using license verification tools, such as Fossology[7] or Black Duck[8], which can automatically detect possible license violations regarding used libraries. The metadata added to the ROAR is used to describe and identify the research software and any data and publications it contains. Using this metadata, the ROAR can be found by other scientist in order to reuse the contained software or data (see step 4). Furthermore, a unique ID can be assigned to a ROAR for enabling the citation of it. Since a ROAR is a standards-based packaging format containing all the necessary components for enabling the automated provisioning (cf. Sect. 2.2), the modeled application can be automatically deployed using a standard-compliant deployment engine, such as the open-source runtime environment *OpenTOSCA Container*[9] [7]. Therefore, scientists do not require any expert IT knowledge at all and are able to use and recreate the research results as often as they want to.

4 Related Work

Packaging software, data, and publications together into one archive as well as enabling the provisioning of software have been in the focus of different research areas. Therefore, in this section, we complete our discussion about related work, which we already discussed partially in Sect. 1.

Weigel et al. [29] give a recommendation for *actionable collections* and a technical interface specification to enable client-server interaction. Their focus lies on the management of research collections. To enable and automate the provisioning of software in different environments, concepts of [16] and [30] have already been developed. Képes et al. [16] presents a template-based concept for provisioning software using a template that is suitable for a specific infrastructure. Zimmermann et al. [30] uses the infrastructure or database components available in an environment to connect the software and generate an overall model for provisioning. These concepts can be used to provision research software in the available environment and to connect it to existing data sources but did not cover the integrated publishing and description of software, data, workflow and licensing of the research results. Stodden et al. [27] provides a platform to store source code and data and run them directly in a cloud environment but did not provide licence checks and an integrated publishing format. For general storage and easy software execution, Boettiger [9] proposes Docker

6 https://projects.eclipse.org/projects/soa.winery
7 https://www.fossology.org/
8 https://www.blackducksoftware.com/
9 https://github.com/OpenTOSCA/container

containers to virtualize the exact system environment in which the original results were produced. The Software Heritage Archive [13] aims to collect all source code that is publicly available. The development history will be archived with an index in order to make the code referenceable and accessible. This approach is an important archive for software source code, however, is not intended to package all artifacts together in order to enable the execution of the software for reuse.

5 Conclusion

In this paper, we illustrated our approach of a comprehensive solution for packaging research software together with data, metadata, license information, and publications in a portable way, which enables reusability as well as reproducibility in science. Therefore, in this work we first presented an overview of the current issues and problems in this area. Furthermore, we presented the concepts of Research Objects and the TOSCA standard and sketched how both concepts can be combined in order to create our approach of Research Object Archives. Moreover, we depicted the composition and the ingredients of a ROAR and characterized the features it provides. We also illustrated how the portability of a ROAR is achieved and how the automated deployment is realized. In future work, we focus on the implementation and integration of the proposed concepts into the TOSCA modeling tool Winery and OpenTOSCA Container.

Acknowledgments This work was partially funded by the projects *SePiA.Pro* (01MD16013F) and *IC4F* (01MA17008G).

References

1. Almeida, D.A., Murphy, G.C., Wilson, G., Hoye, M.: Do Software Developers Understand Open Source Licenses? In: ICPC. pp. 1–11. IEEE (2017)
2. Barnes, N.: Publish your computer code: it is good enough. Nature News 467(7317), 753–753 (2010)
3. Bechhofer, S., Buchan, I., De Roure, D., Missier, P., Ainsworth, J., Bhagat, J., Couch, P., Cruickshank, D., Delderfield, M., Dunlop, I., Gamble, M., Michaelides, D., Owen, S., Newman, D., Sufi, S., Goble, C.: Why linked data is not enough for scientists. Future Generation Computer Systems 29(2), 599–611 (2013)
4. Bechhofer, S., De Roure, D., Gamble, M., Goble, C., Buchan, I.: Research Objects: Towards Exchange and Reuse of Digital Knowledge. In: FWCS. Nature Precedings (2010)
5. Belhajjame, K., Zhao, J., Garijo, D., Gamble, M., Hettne, K., Palma, R., Mina, E., Corcho, O., Gómez-Pérez, J.M., Bechhofer, S., Klyne, G., Goble, C.: Using a suite of ontologies for preserving workflow-centric research objects. Web Semantics: Science, Services and Agents on the World Wide Web 32, 16–42 (2015)
6. Bergmayr, A., Breitenbücher, U., Ferry, N., Rossini, A., Solberg, A., Wimmer, M., Kappel, G., Leymann, F.: A Systematic Review of Cloud Modeling Languages. ACM Computing Surveys (CSUR) 51(1), 22:1–22:38 (2018)

7. Binz, T., Breitenbücher, U., Haupt, F., Kopp, O., Leymann, F., Nowak, A., Wagner, S.: OpenTOSCA - A Runtime for TOSCA-based Cloud Applications. In: ICSOC. pp. 692–695. Springer (2013)
8. Binz, T., Breitenbücher, U., Kopp, O., Leymann, F.: TOSCA: Portable Automated Deployment and Management of Cloud Applications, pp. 527–549. Advanced Web Services, Springer (2014)
9. Boettiger, C.: An introduction to docker for reproducible research. ACM SIGOPS Operating Systems Review 49(1), 71–79 (2015)
10. Breitenbücher, U., Binz, T., Kopp, O., Leymann, F.: Vinothek - A Self-Service Portal for TOSCA. In: ZEUS. pp. 69–72. CEUR-WS.org (2014)
11. Breitenbücher, U., Binz, T., Kopp, O., Leymann, F., Wettinger, J.: Integrated Cloud Application Provisioning: Interconnecting Service-Centric and Script-Centric Management Technologies. In: CoopIS. pp. 130–148. Springer (2013)
12. Collberg, C., Proebsting, T., Warren, A.M.: Repeatability and benefaction in computer systems research. University of Arizona TR 14-4 (2015)
13. Cosmo, R.D., Zacchiroli, S.: Software Heritage: Why and How to Preserve Software Source Code. In: iPRES (2017)
14. Endres, C., Breitenbücher, U., Falkenthal, M., Kopp, O., Leymann, F., Wettinger, J.: Declarative vs. Imperative: Two Modeling Patterns for the Automated Deployment of Applications. In: Proceedings of the 9th International Conference on Pervasive Patterns and Applications (PATTERNS). pp. 22–27. Xpert Publishing Services (2017)
15. Hunter, J.: Scientific publication packages–A selective approach to the communication and archival of scientific output. International Journal of Digital Curation 1(1), 33–52 (2008)
16. Képes, K., Breitenbücher, U., Leymann, F.: The SePaDe System: Packaging Entire XaaS Layers for Automatically Deploying and Managing Applications. In: CLOSER. pp. 626–635. SciTePress (2017)
17. Kopp, O., Binz, T., Breitenbücher, U., Leymann, F.: BPMN4TOSCA: A Domain-Specific Language to Model Management Plans for Composite Applications. In: Proceedings of the 4th International Workshop on the Business Process Model and Notation (BPMN 2012). pp. 38–52. Springer (2012)
18. Kopp, O., Binz, T., Breitenbücher, U., Leymann, F.: Winery – A Modeling Tool for TOSCA-based Cloud Applications. In: ICSOC. pp. 700–704. Springer (2013)
19. Kopp, O., Binz, T., Breitenbücher, U., Leymann, F., Michelbach, T.: A Domain-Specific Modeling Tool to Model Management Plans for Composite Applications. In: Proceedings of the 7th Central European Workshop on Services and their Composition, ZEUS 2015. pp. 51–54. CEUR Workshop Proceedings (2015)
20. Nyström, P., Falck-Ytter, T., Gredebäck, G.: The TimeStudio Project: An open source scientific workflow system for the behavioral and brain sciences. Behavior Research Methods 48(2), 542–552 (2016)
21. OASIS: Web Services Business Process Execution Language (WS-BPEL) Version 2.0. Organization for the Advancement of Structured Information Standards (OASIS) (2007)
22. OASIS: Topology and Orchestration Specification for Cloud Applications (TOSCA) Primer Version 1.0. OASIS (2013)
23. OASIS: Topology and Orchestration Specification for Cloud Applications (TOSCA) Version 1.0. OASIS (2013)
24. OASIS: TOSCA Simple Profile in YAML Version 1.0. OASIS (2015)
25. OMG: Business Process Model and Notation (BPMN) Version 2.0. Object Management Group (OMG) (2011)

26. Pizzi, G., Cepellotti, A., Sabatini, R., Marzari, N., Kozinsky, B.: Aiida: automated interactive infrastructure and database for computational science. Computational Materials Science 111, 218–230 (2016)
27. Stodden, V., Hurlin, C., Pérignon, C.: Runmycode.org: A novel dissemination and collaboration platform for executing published computational results. In: eScience. pp. 1–8. IEEE (2012)
28. Stodden, V., McNutt, M., Bailey, D.H., Deelman, E., Gil, Y., Hanson, B., Heroux, M.A., Ioannidis, J.P., Taufer, M.: Enhancing reproducibility for computational methods. Science 354(6317), 1240–1241 (2016)
29. Weigel, T., Almas, B., Baumgardt, F., Zastrow, T., Schwardmann, U., Hellström, M., Quinteros, J., Fleischer, D.: Recommendation on Research Data Collections. Research Data Alliance (2017)
30. Zimmermann, M., Breitenbücher, U., Falkenthal, M., Leymann, F., Saatkamp, K.: Standards-based Function Shipping – How to use TOSCA for Shipping and Executing Data Analytics Software in Remote Manufacturing Environments. In: EDOC. pp. 50–60. IEEE (2017)