DevOps: Introducing Infrastructure-as-Code

Matej Artač[2], Tadej Borovšak[2], Elisabetta Di Nitto[1], Michele Guerriero[1], Damian Andrew Tamburri[1]
[1]DEEPSE group - DEIB - Politecnico di Milano, Milano, Italy
[2]XLAB - Ljubljana, Slovenia
matej.artac@xlab.si, tadej.borovsak@xlab.si, elisabetta.dinitto@polimi.it,
michele.guerriero@polimi.it, damianandrew.tamburri@polimi.it

***Abstract*—DevOps entails a series of software engineering tactics aimed at shortening the actionable operation of software design changes. One of these tactics is to harness infrastructure-as-code, that is, writing a blueprint that contains deployment specifications ready for orchestration in the cloud. This abstract briefly discusses all necessary elements and abstractions in writing and maintaining that blueprint, revolving around a key standard for its expression, namely, the OASIS "Topology and Orchestration Specification for Cloud Applications" (TOSCA) industrial standard adopted by as many as 60+ big industrial players worldwide.**

***Keywords*-DevOps; Infrastructure-as-Code; TOSCA;**

I. Introduction

The current IT market is increasingly dominated by the "need for speed". This need is reflected in the emerging use of agile and lean techniques which shorten the software development cycle and also intermix software development activities with IT operations [1]. This trend of using software engineering tactics that reduce the space, time and efforts between software development and operations as well as the technical and organizational distance between these two types of software teams is known as DevOps [1]. As part of the DevOps menu, many practices entail re-using standard tools from software development (e.g., code-versioning, code-revision management, etc.) to manage what is known as *infrastructure-as-code* (IasC) [2]. IasC promotes managing knowledge and experience of plethora of subsystems as a single commonly available source of truth instead of traditionally reserving it for system administrators. This briefing introduces all essential technologies involved in supporting infrastructure-as-code, starting from the industrial standard for IasC, OASIS TOSCA, which stands for "Topology and Orchestration Specification for Cloud Applications" [3] and elaborating more on connected technologies, such as deployment blueprints or TOSCA-ready orchestration engines. TOSCA is state-of-the-art industrial practice language for automated deployment of technology independent and multi-cloud compliant applications, supported by players such as HP, CISCO Systems, Cloudera, Huawei, IBM [3].

The following contains a brief overview of DevOps, addressing TOSCA and IasC together.

II. DevOps in Pills

In essence the core idea behind DevOps is fairly simple: over long years of "siloed" [4] software development (on one side) and software operations (on the other), a series of distances in technical, socio-technical, and organizational practices and tools has been flourishing between Dev- and -Ops teams [1]. These distances make the fluent and coherent communication, collaboration and issue resolution between the two silos practically difficult if not impossible. DevOps entails a series of socio-technical and organizational practices and tools that address and tackle these distances one by one until a *frictionless*, fluent, and resilient organization is distilled. Of the several socio-technical practices under study we mention, most prominently, the organizational blending of Dev- and -Ops teams under the same working unit with consequent shifting of operational concerns (think of, for example, infrastructure design, dimensioning, cost-optimization [5]) to be addressed by the entire team, including developers. Other similar socio-technical and organizational practices lean towards the shared use of "DevOps" tools [6], i.e., tools that promote the use of the same knowledge-base (think of an issue-tracking mechanism) for both development and operation such that there shall be no misunderstanding or knowledge gap between developer and operator knowledge at all.

III. Introducing Infrastructure-as-Code

Infrastructure design is the software lifecycle phase that defines and configures the software infrastructure needs for that software, e.g., the AWS cloud, the number and type of VMs required, etc. Infrastructure designs typically entail many installation and configuration scripts needed to, among others: (a) instantiate and link the required machines (either physical or virtual) for the software to run; (b) install and configure the required software and middleware for those VMs; (c) instantiate and run the needed ancillary services for the software to be operated. In the scope of these designs, DevOps promotes the use of a typical software development notion, i.e., "source code", for infrastructure designs as well, such that the entire set of scripts, automation and configuration code, models, required dependencies and operational configuration parameters can be expressed using the same standard language, much like you would use Java and ancillary Java libraries to put together your own software application - this practice is *infrastructure-as-code*. The purpose behind this proposal is to re-use successful and common software development practices to speed up software operations: think of using infrastructure code versioning for the purpose of de-bugging and back-tracking/version control or using infrastructure design

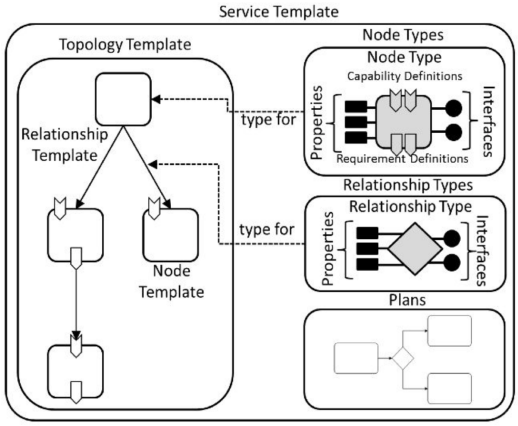

Fig. 1: TOSCA - nodes and edges.

patterns to quickly put together common solutions to known problems [1] or even exploiting model-driven engineering to pull IasC specifications immediately from architectural design and development models [7]. In fact, IasC is a key enabler of several DevOps tenets that heavily depend on automation. For example, continuous testing is made drastically easier if a single IasC code or *blueprint* allows you to pull up a canary-test environment in which to experiment with your application [8]. Similarly, continuous architecting scenarios may focus on improve the infrastructure code just as much as improving the architecture designs as well.

IV. Where does TOSCA fit in?

As previously stated, TOSCA allows to design and program infrastructure designs using a standard notation and language. TOSCA provides reusable nodes and edges (see Fig. 1) to specify an infrastructure as a graph, or *topology*, using strong-typing to speed-up reuse. Quoting from the standard specification itself [3], "TOSCA [...] uses [...] service templates to describe cloud apps as a topology template, [...]. TOSCA provides a type system of node types to describe the possible building blocks for constructing a service template, as well as relationship type to describe possible relations". TOSCA-ready orchestrators such as Cloudify[1] and Apache Brooklyn [2] normally come with a considerable number of reusable TOSCA nodes (e.g., a MySQL DB, a WordPress host, etc.), while more are proliferating in both research and practice.

V. Putting it all together: the DICER example

As an example of what can be done putting TOSCA and IasC in action, we briefly outline DICER (Data-Intensive Continuous Engineering and Rollout) [9], a model-driven tool that allows to quickly put together the infrastructure design for a big data cloud application as part of continuous data-intensive architecting [10]. In a nutshell, DICER comprises two main components, the *Modelling Environment* (realised using EMF and Eclipse GMF generative technologies) and the *Deployment Service* (a tailored ad-hoc version of the Cloudify TOSCA-ready orchestrator). Via the Modelling Environment a user (e.g., an infrastructure engineer) can drag and drop deployment nodes directly from a familiar Eclipse IDE modelling interface, properly linking them using OCL-assisted dependency checking and configuring them according to nodes typical properties (e.g., VM replication factor for VM hosts, etc.). Once the user is satisfied with the infrastructure design, DICER allows 1-click deployment of TOSCA blueprints that reflect the user's model, using its own deployment service.

[1]http://getcloudify.org/

[2]https://brooklyn.apache.org/

VI. Conclusion

This abstract briefly outlines DevOps with a specific focus on infrastrcture-as-code as a tactic to accelerate DevOps-based lifecycles and TOSCA as the de-facto standard to specify IasC. In our mid- and long-term agenda we envision further researching and using this tactic from an experimental and empirical software engineering perspective, furthering our understanding of how common software development tactics can be reapplied to infrastructure design.

Acknowledgment

The authors' work is partially supported by the European Commission grant no. 644869 (H2020 - Call 1), DICE.

References

[1] L. J. Bass, I. M. Weber, and L. Zhu, *DevOps - A Software Architect's Perspective.*, ser. SEI series in software engineering. Addison-Wesley, 2015.

[2] K. Morris, *Infrastructure As Code: Managing Servers in the Cloud*. Oreilly & Associates Incorporated, 2016. [Online]. Available: https://books.google.si/books?id=kOnurQEACAAJ

[3] P. Lipton, D. Palma, M. Rutkowski, and D. A. Tamburri, "Tosca solves big problems in the cloud and beyond!" *IEEE Cloud*, vol. 21, no. 11, pp. 31–39, 2016.

[4] D. A. Tamburri, P. Kruchten, P. Lago, and H. van Vliet, "Social debt in software engineering: insights from industry." *J. Internet Services and Applications*, vol. 6, no. 1, pp. 10:1–10:17, 2015. [Online]. Available: http://dblp.uni-trier.de/db/journals/jisa/jisa6.html#TamburriKLV15

[5] M. Ciavotta, D. Ardagna, and G. P. Gibilisco, "A mixed integer linear programming optimization approach for multi-cloud capacity allocation." *Journal of Systems and Software*, vol. 123, pp. 64–78, 2017. [Online]. Available: http://dblp.uni-trier.de/db/journals/jss/jss123.html#CiavottaAG17

[6] M. Hüttermann, *DevOps for Developers*. New York: Apress, 2012.

[7] M. Guerriero, S. Tajfar, D. A. Tamburri, and E. D. Nitto, ""towards a model-driven design tool for big data architectures"," in *proceedings of the 2nd International Workshop on Big Data Software Engineering - BIGDSE*. IEEE, 2016.

[8] S. Johann, "Kief morris on infrastructure as code." *IEEE Software*, vol. 34, no. 1, pp. 117–120, 2017. [Online]. Available: http://dblp.uni-trier.de/db/journals/software/software34.html#Johann17

[9] M. Artac, T. Borovsak, E. D. Nitto, M. Guerriero, and D. A. Tamburri, "Model-driven continuous deployment for quality devops." in *QUDOS@ISSTA*, D. Ardagna, G. Casale, A. van Hoorn, and F. Willnecker, Eds. ACM, 2016, pp. 40–41. [Online]. Available: http://dblp.uni-trier.de/db/conf/issta/qudos2016.html#ArtacBNGT16

[10] E. D. Nitto, P. Jamshidi, M. Guerriero, I. Spais, and D. A. Tamburri, "A software architecture framework for quality-aware devops." in *QUDOS@ISSTA*, D. Ardagna, G. Casale, A. van Hoorn, and F. Willnecker, Eds. ACM, 2016, pp. 12–17. [Online]. Available: http://dblp.uni-trier.de/db/conf/issta/qudos2016.html#NittoJGST16