A model-driven approach for DevOps

Hugo da Gião
University of Porto, Faculty of Engineering & HASLab/INESC TEC, Portugal
hugo.a.giao@inesctec.pt

***Abstract*—DevOps is a combination of tools and methodologies that aims at improving the software development, build and deploy processes by shortening this lifecycle and improving software quality. Despite its many benefits it still presents many challenges when it comes to its ease of use and accessibility. One of the reasons is the tremendous proliferation of different tools, languages and syntax which makes the field quite difficult to learn and keep up to date.**

Our goal with this PhD project is to study a model-driven approach that can abstract the particularities of the different kinds of tools for the different parts of the DevOps process (from source code management to deployment). Moreover, we will also use model-driven techniques to allow users to express their pipelines without the need to know all the implementation details (e.g. configuration files) of all the tools they need to use.

***Index Terms*—DevOps, end-user software development, model-driven engineering**

I. Introduction

DevOps is a means for software development and delivery by using various tools and techniques that integrate the worlds of development and operations [1]. The DevOps word comes from the combination of Development with Operations. Indeed, this term is used to describe a culture where these two realms are no longer separated as in the past, but are now just one. The integration of these two fields is achieved through automated development, build, deployment, and monitoring. One of the goals is to achieve reliable, secure, fast and continuous delivery of software, to improve productivity in businesses and workers' well being. Moreover, software engineers that use DevOps are more resilient and equipped for fast changes [2]. However organizations and developers face many obstacles when adopting DevOps, including changes in the architectural organization, dealing with problems in older infrastructure and the integration of different DevOps tools [3]. Other problems faced by developers are in learning and using the specific DevOps tools which can be overwhelming and can decrease the improvements expected with the adoption of DevOps [4].

Model-driven engineering (MDE) has been used to cope with numerous problems in computer science, from complexity to the increase of compatibility between different systems, to aid in the design process, or in improving the communication between different teams and team members [5]. Models are in many cases visual languages, or can be represented as such, and we believe they are a good approach to remove the complexity associated with DevOps, making it easier to adopt and use. Thus, in this project we intend to use MDE to cope with the complexity of DevOps and to make it accessible to more users and in particular users with a less strong background in this kind of technology.

II. State of the Art

Little work has been done by the research community to aid DevOps teams in being more productive. One exception is the work of Piedade et al. which present on the very few approaches to try to improve Docker Compose[1] usability by proposing a visual language that reduces the complexity and learning curve to use this tool by equipping users with an intuitive and functional interface [6].

Some projects have already used MDE for DevOps most notably to represent the DevOps process and software architectures [7], [8], [9], [10]. Despite applying MDE to this process their approaches most of these often focus on the management side of software engineering and do not offer a concrete way of replacing or complementing the use of textual tools. Nevertheless some projects such as ARGON generate and replace textual infrastructure, using the high level specification, more precisely this tool uses UML languages to generate scripts to automate and manage configuration management tools and lets users express cloud deployment operations in a generic approach that can be used for different providers [11].

III. Objectives

In this PhD project our initial goal is to better study and understand some of the problems faced by software engineers and business when adopting and using DevOps. We will do so by reaching out to software companies and interviewing the engineers. We will also discuss with practitioners possible future directions for DevOps so we can try to cope with them in our proposals.

Given the proliferation of tools, languages, and syntaxes related to DevOps, there is a need for improvement in current DevOps adoptions. One of our goals is to create abstract layers capable of automating portions of the DevOps' process and facilitate the developers' work. Thus, we will study model-driven approaches to cope with the diversity in this field.

Beyond improving single steps of the DevOps process we also have the goal of unifying these steps into an integrated solution that can facilitate the whole process for organizations

[1]Docker Compose (https://docs.docker.com/compose) is a tool to define and run multi container applications, being itself already an improvement over the use of Docker (https://docs.docker.com), a tool for managing containers.

978-1-6654-4214-5/22/$31.00 ©2022 IEEE

and developers alike. Thus, we need to study how to integrate the different steps in a pipeline that can be instantiated by each team for each project.

Thus, during this PhD project we will answer the following research questions:

RQ1: How has DevOps evolved over time in practice and where might it go in the future?

RQ2: To which extent is it possible to design a language that is capable of abstracting the current DevOps different parts and cope with future changes and technologies?

RQ3: What is the best methodology to integrate the different parts of DevOps and turn it into a unified process?

IV. Our approach

The DevOps process can be divided in 3 parts: i) development, ii) build management, and iii) deployment management [1]. In this PhD project we will address each of these parts as well as their integration.

A. Analysis and study of the DevOps process

Our initial goal is to approach industrial partners and find the different techniques used to implement DevOps as well as the most notable problems companies have in their daily lives. We will meet with partner companies' DevOps teams using semi-structured interviews [12] to gather initial industrial evidence of the current trends in DevOps. Moreover, we will also try to understand previous processes they had and what they think the future might be so we can understand the evolution of DevOps in practice and prepare our methodologies for future changes. Although we will start with these two companies, we will also contact others to collect significant information about the state-of-the-art of DevOps in industrial context. This work will give us plenty of information so we can make decisions based on empirical evidence.

B. Design and implementation of a generic framework for the DevOps process

DevOps is characterized by an enormous amount of tools available for each particular task. This implies DevOps teams need to learn quite some tools, languages, notations to be able to define concrete DevOps processes. This is quite challenging given the amount of tools, not stable as tools are created/changed frequently, and difficulty to integrate new members as there are too many concepts to be learned before starting to work. Thus, our goal is to create a framework capable of specifying the different parts of the DevOps process independently of the concrete technological choices of each DevOps team.

To this end we will study model-driven methodologies and propose a meta-model that can be used to generate different models for the three parts of the process we mentioned before [1]. This meta-model will need to include concepts from development, build management and deployment. Each of these models can then be instantiated in the different tools. For instance, given a model for build management, one can instantiate it in tools such as Gradle or Ant. The intent is that our framework can generate the necessary code or configuration files based on the model for existing tools, or even for tools that may appear in the future, being only necessary to create the necessary mapping between the model and the new tool's concrete syntax.

Such a framework will allow DevOps teams to have a single specification of the different parts of the process and generate concrete instances of the tools they choose for each project or even in different periods of time.

Each of the models and the meta-model will be validated using case studies collected in the initial part of this project. This validation will be mostly an applicability validation, that is, we will evaluate if the models are sufficient to represent a significant amount of tools. After this validation we can create a meta-model. These (meta-)models will then be implemented using a framework such as Epsilon [13] and empirically validated with DevOps teams.

C. DevOps pipelines without concrete technologies in mind

In this part of the work we will study how to integrate the different steps of the DevOps process in a unified process. Each team has their own way to integrate the different parts of the process and thus we need to find a way to allow them to define arbitrary combinations of the DevOps steps.

We will study how to extend the previously proposed models to include concepts from pipelines. We will also study other languages such as the Business Process Modeling Notation (BPMN) which is quite used in different industrial contexts and thus may be more appealing for industrial practitioners.

Once more, the language we will propose will be validated for its completness by applying it to several case studies. Nevertheless, we will evaluate it empirically with practitioners so we can assess its usability (efficiency, effectiveness and satisfaction) by real users.

V. Concluding remarks

The use of DevOps is currently quite standard in industry. However, despite its benefits, it poses quite some challenges as more as more tasks are asked to DevOps teams. In this PhD project we will study how to use MDE to cope with this growing complexity and how MDE can aid DevOps teams being more effective and efficient.

Acknowledgments

This work is financed by the ERDF - European Regional Development Fund, through the Operational Programme for Competitiveness and Internationalisation - COMPETE 2020 Programme under the Portugal 2020 Partnership Agreement, and by National Funds through the FCT - Portuguese Foundation for Science and Technology, I.P. on the scope of the UT Austin Portugal Program within project BigHPC, with reference POCI-01-0247-FEDER-045924.

References

[1] C. Ebert, G. Gallardo, J. Hernantes, and N. Serrano, "Devops," *IEEE Software*, vol. 33, no. 3, pp. 94–100, 2016.

[2] G. Kim, J. Humble, P. Debois, and J. Willis, *The DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations*, ser. ITpro collection. IT Revolution Press, 2016. [Online]. Available: https://books.google.pt/books?id=ui8hDgAAQBAJ

[3] G. B. Ghantous and A. Q. Gill, "Devops: Concepts, practices, tools, benefits and challenges," in *PACIS*, 2017.

[4] M. U. Haque, L. H. Iwaya, and M. A. Babar, "Challenges in docker development: A large-scale study using stack overflow," in *Proceedings of the 14th ACM / IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, ser. ESEM '20. New York, NY, USA: Association for Computing Machinery, 2020. [Online]. Available: https://doi.org/10.1145/3382494.3410693

[5] D. Schmidt, "Guest editor's introduction: Model-driven engineering," *Computer*, vol. 39, no. 2, pp. 25–31, 2006.

[6] B. Piedade, J. a. P. Dias, and F. F. Correia, *An Empirical Study on Visual Programming Docker Compose Configurations*. New York, NY, USA: Association for Computing Machinery, 2020. [Online]. Available: https://doi.org/10.1145/3417990.3420194

[7] A. Colantoni, L. Berardinelli, and M. Wimmer, *DevOpsML: Towards Modeling DevOps Processes and Platforms*. New York, NY, USA: Association for Computing Machinery, 2020. [Online]. Available: https://doi.org/10.1145/3417990.3420203

[8] F. Bordeleau, J. Cabot, J. Dingel, B. S. Rabil, and P. Renaud, "Towards modeling framework for devops: Requirements derived from industry use case," in *Software Engineering Aspects of Continuous Development and New Paradigms of Software Production and Deployment*, J.-M. Bruel, M. Mazzara, and B. Meyer, Eds. Cham: Springer International Publishing, 2020, pp. 139–151.

[9] L. Burgueño, F. Ciccozzi, M. Famelis, G. Kappel, L. Lambers, S. Mosser, R. F. Paige, A. Pierantonio, A. Rensink, R. Salay, G. Taentzer, A. Vallecillo, and M. Wimmer, "Contents for a model-based software engineering body of knowledge," *Software and Systems Modeling*, vol. 18, no. 6, pp. 3193–3205, 12 2019. [Online]. Available: https://doi.org/10.1007/s10270-019-00746-9

[10] J. Cabot. [Online]. Available: https://modeling-languages.com/devops-modeling-workshop/

[11] J. Sandobalín, E. Insfran, and S. Abrahão, *ARGON: A Tool for Modeling Cloud Resources*, 06 2018, pp. 393–397.

[12] C. Wohlin, P. Runeson, M. Hst, M. C. Ohlsson, B. Regnell, and A. Wessln, *Experimentation in Software Engineering*. Springer Publishing Company, Incorporated, 2012.

[13] L. Tratt and M. Gogolla, *Theory and Practice of Model Transformations*, 01 2010, vol. 6142.