PERFORMANCE ANALYSIS OF DEVOPS BASED HYBRID MODELS INTEGRATED WITH DIFFERENT AUTOMATION TOOL CHAINS FOR QUALITY SOFTWARE DEVELOPMENT

Poonam Narang, Pooja Mittal*

Department of Computer Science and Applications,
Maharshi Dayanand University, Rohtak, Haryana
India

* Corresponding Author: e-mail: poonam.mehta20@gmail.com

Abstract: Software development methodologies have evolved through traditional waterfall model to recent DevOps development culture. This research validates the performance of our previously proposed, implemented and verifed DevOps-based hybrid model of integrated automation tools. For this purpose, underlying work considers random DevOps automation tool chain model based on current market and industry scenario followed by the evaluation of software quality metrics for both models and conclude comparative results in the form of tables. The results confirm that a randomly selected tool chain performs better, but if the tool chain is pre-determined or fixed, DevOps performance improves many folds, which validates the performance of our DevOps-based hybrid model of integrated automation tools. This research will be useful for software developers to select best automation tools that not only speed up the development process but also deliver quality software.

Key words: Automation, Automation Tools, DevOps, Software Development, Tool Chain.

1. INTRODUCTION

In a survey of software tools, methods and results [1], many problems were encountered in different phases of software development life cycle. Development of large scale software systems were very much unpredictable and difficult in earlier times. Engineering discipline was in need of more scientific theory of development instead of trial and error paradigm [2]. The use of complex hardware systems, even for similar types of problems, was a major factor in the demise of traditional methodologies [3, 4]. Many large-scale agile methods comparison [5, 6] and results of ethnographic study [7], found that Agile methods of software development, are in need of complete automation and continuous performance assessment or continuity was also expected in agile methods to improve their performance management. Another literature survey [8, 9] found that the effects of continuity in terms of integration on software development were

positive, such as improved team cooperation. To learn, assess, verify and validate continuous environment in software development, our research focuses around DevOps, latest buzzword in the development industries that promises to deploy successful quality projects with the use of five Cs. These Cs of continuous phases include continuous integration, continuous testing, continuous delivery, continuous deployment and continuous monitoring. DevOps employs different set of automation tools at each and every phase of software development. Research on automation tools at all stages of DevOps, including testing [10, 11], as well as another empirical study on DevOps adoption trends [12], confirm the use of automation to solve a wide range of real-world problems.

In this context, this study considers quantitative evaluation of our previously proposed [13], implemented [14] and verified [15] DevOps-based hybrid model with integrated automation tools and compares it to another DevOps model with randomly selected set of automation tools, chosen on the basis of current market demand and common usage scenario. The purpose of this evaluation is to validate the performance of earlier implemented tool chain models based on metric measurement and comparison. For performing metric calculations, the same code repositories [13] have been taken as sample projects that, were implemented with another set of automation tools. This research considers Teamcity, Jmeter, GitHub Actions, Puppet, and Chef as different automation tools set for five continuous stages of DevOps. Better result outcome again validates the out performance of underlying DevOps-based hybrid model of selective and integrated automation tools. The results of this research will be of big advantage to developers or researchers to get in depth knowledge about DevOps automation tool set. It will also be useful in terms of automated tool chain selection and reduction of development time up to much greater extent. Our young students will also benefit in understanding the latest technology in software development industries. Other sections of the research paper are- literature review, DevOps working principles, proposed and implemented DevOps-based hybrid model followed by implementation and result discussion with another tool chain set for comparative analysis. Last section of the paper concludes with the discussion of future work.

2. LITERATURE REVIEW

IT industries have been shifting their software development paradigm from traditional methodologies to the more recent development culture of DevOps for many years. Many renowned researchers' work is currently being studied in the literature, beginning with traditional development models and gradually progressing to recent DevOps applications. Winston W. Royce [16] described the waterfall model in his paper as a step-by-step development at different iterative stages. The author supports these approaches for large-scale system development, but acknowledges the inflexible and aggressive nature of this development model. Many other authors, including [17] and [18], discuss agile development methods and a set of practices that can easily adapt to changing customer needs. Agile methods overcome all constraints of traditional development models and are well suited for both small and large scale development, teams and projects [18, 19]. Authors also conducted an online survey on agile limitations,

with DevOps as a proposed solution to improve software quality [20, 21]. Besides traditional and Agile methodologies, DevOps as the latest technology or development approach involves alternative set of automation tools to touch the targets of quality, speedy and in-budget delivery of software. As clarified in research work, DevOps, with the support of its automation tools and continuous development and operations environment, eases down the path of software development industries with the increase quality rapid releases [21, 22]. Different case studies [23] also confirms the lack of metric measurements and gaps in DevOps applications [24-26].

Motivation behind this work is to validate our implemented and verified proposed tool chain of DevOps-based hybrid model integrated with different automation tools by comparative analysis with another randomly selected model of tool chain. In terms of reduced development time and quality deployment, a pre-selected tool chain outperforms a random tool selection.

3. DEVOPS-BASED HYBRID MODELS OF INTEGRATED TOOL CHAIN (ITC)

For the quality validation of DevOps automation tools, we have considered our proposed [13], implemented [14] and verified [15] DevOps-based hybrid model consisting of integrated tool chain (ITC) as depicted in figure 8 of [13]. For the hybrid model, different representative automation tools at each continuous phase of DevOps are chosen based on current market demands, industry usage scenario along with google and stack overflow recent trends. These representatives were compared analytically on the basis of different performance evaluators and best performing tools are included in DevOps-based hybrid model of tool chain. Our verified model already performs better as compared to traditional software development methodologies [15]. However, in order to validate our model, the current paper compares it to a random chain of tools in order to find a tool chain that provides a better solution for developers in terms of delivery and deployment time reduction.

4. METHODOLOGY

For the validation of DevOps-based hybrid model, we have considered three sample java-based projects defined in figure 1 of [15]. These projects or code are created in local JDK repositories and then uploaded to GitHub to become a remote repository, to ensure the smooth operation of the DevOps environment. The flow of research work is shown in figure 1 in terms of writing java-based code followed by their implementation in DevOps environment consisting of different tool chains.

Tools like Jenkins and TeamCity, allows the developers to continuously integrate updated code modules with central repositories in order to maintain more code stability. Similarly, Jmeter is responsible for automated scheduling of testing after every code update. GitHub Actions is selected for the purpose of another integrated tool chain comparison. Puppet is termed as best Continuous deployment tool after Ansible that is used for immediate automated code deployment to the production environment. Sensu is chosen as another best performer continuous monitoring tool after Nagios to allow frequent changes in user needs.

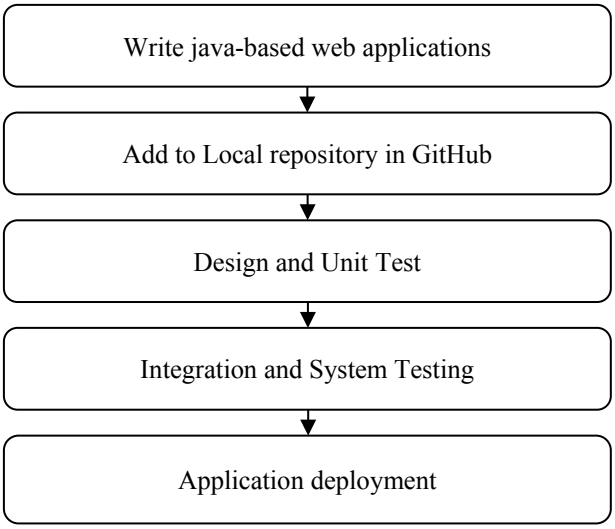

Figure 1. Research workflow for project implementation through DevOps-based models of automation tools

5. METRIC SELECTION WITH RESULTS AND DISCUSSIONS

Software metric evaluation is an essential requirement for the measurement of project progress, successful delivery, deployment and operations of the whole process, project and product. Different metrics defined in our previously implemented DevOps-based hybrid model were earlier evaluated for DevOps and Traditional development with java-based projects. These sample projects mentioned in [15] have Java as application development environment with Eclipse IDE and Tomcat Server is linked for deployment. Results obtained through metric measurement, verified the performance of DevOps-based hybrid model of selected automation tools [15]. These metrics are reconsidered for evaluation against the same data set using DevOps-based hybrid model and another randomly chosen tool chain selected on the basis of market demand and usage.

5.1. Project Defect Density (PDD)

Project defect density is directly dependent on the presence of defects in the system [15]. Defect density formula is given as

$$\text{PDD} = \sum_{1=1}^{n} \frac{TotalNumberofDefects}{SizeofSoftware\ (in\ KLOC)} \quad (1)$$

Table 1 displays the total defect density of the project or system as calculated using DevOps-based hybrid model of selective and integrated tool chain [15] as Mod1 along with results obtained from randomly selected tool chain as Mod2.

The results clearly show that the already implemented tool chain performs better in terms of low defect density when compared to the results of the next considered automation tool chain set.

Table1. PDD Measure of sample projects as per Implemented hybrid model and another Tool Chain Set

Sr No	Project	Size (LOC)	No of Components/ Modules	Total no of Defects		Defect Density (PDD)	
				Mod1	Mod2	Mod1	Mod2
1	Project1	2430	18	15	22	6.17	9.05
2	Project2	1579	14	8	15	5.07	9.50
3	Project3	557	8	4	6	7.18	10.77

5.2. Release Deployment Frequency (RDF)

The total number of deployments in a given time period is indicated by the release deployment frequency [15]. The formula for calculating deployment frequency is as follows.

$$\text{RDF} = \sum_{i=1}^{n} \frac{TotalNumberofDeployments}{TimeUnits\ (inHours)} \quad (2)$$

Table 2 displays data on the frequency of deployment of Java projects using both pre-selected and randomly selected chain of automation tools.

Table2. RDF Measure of sample Projects using Proposed ITC OF Hybrid Model

Sr No	Project	Size (LOC)	No of deployments		Time taken to deploy (hr)		Deployment Frequency (RDF)	
			Mod1	Mod2	Mod1	Mod2	Mod1	Mod2
1	Project1	2430	20	20	1.1	1.3	18.18	15.38
2	Project2	1579	16	16	0.9	1.1	17.78	14.55
3	Project3	557	10	8	0.7	0.9	14.29	8.89

Results clearly indicate the high value of deployments that leads to acceptance to frequent changes to the system.

5.3. System Risk Identification (SRI)

System risk identification refers to the assessment of risk associated with the system to be developed [15].Expression or formula for system risk identification is given as

$$\text{SRI} = \sum_{i=1}^{n} Wx, \qquad n>0 \quad (3)$$

Wx denotes the weightage assigned to each individual risk and the total number of components in the system is denoted by n. Table 3 depicts risk identification measurement for Mod1 and Mod2 models of DevOps tool chains.

Table 3 results again clearly indicate the high value of risk coverage as compared to model 2 of automation tools.

Table 3. RI Estimate of sample Applications using implemented tool chain set

Sr No	Project	Total Risk Tests	Total no of risk tests executed		Risks broken		Total Risks not tested		Risks not executed		Risk Coverage (%age)	
			Mod1	Mod2	Mod1	Mod2	Mod1	Mod2	Mod1	Mod2	Mod1	Mod2
1	Project1	275	211	210	23	20	24	23	17	22	76.73	76.34
2	Project2	150	116	110	12	10	13	11	9	19	42.18	40
3	Project3	60	46	40	5	5	5	5	4	10	16.62	14.55

5.4. Process Productivity (PP)

The total units of work completed in a given time period is the productivity of any system. [15] Productivity of a process or system is measured as eq. 4.

$$PP = \sum_{i=1}^{n} \frac{TotalNumberofUserStories}{TimeTakenToComplete} \quad (4)$$

Table 4 shows the PP for already proposed and implemented tool chains.

Table 4. PP Estimate/ Measure of current data set using Proposed Tool Chain

Sr No	Project	Size (LOC)	Total No of user stories	Time taken (in weeks)		Process Productivity (PP)	
				Mod1	Mod2	Mod1	Mod2
1	Project1	2430	180	14	15	12.86	12
2	Project2	1579	117	9	10	13	11.7
3	Project3	557	41	3	4	13.67	10.25

Table 4 clearly shows a higher throughput or productivity measure for a DevOps culture that has been implemented. Each of the performance metric shows DevOps as a symbol of quality and speedy delivery with less defect density, good productivity and high coverage of risk sets.

6. CONCLUSION

This study has done quality validation of DevOps-based hybrid model of integrated tool chain over random selection of tool chain. To confirm the validity of DevOps quality, three java projects from different domains are considered along with DevOps performance metrics. For these projects, software components with varying parameters were observed, and values for the metrics under consideration were calculated. The tabular results confirm the speed, quality validation, and decreased development failures with DevOps pre-selected best performer tools over other random tool selections. This carefully selected, already implemented, and verified tool chain validates accelerated development while ensuring high-quality software delivery. Real-time industry data can be collected as part of future work, or other tools to achieve more automated results can be considered.

REFERENCES

[1] Beck, L. L., and Perkins, T. E. A Survey of Software Engineering Practice: Tools, Methods, and Results. *IEEE Transactions on Software Engineering*, SE-9(5), 1983, pp 541–561. http://doi.org/doi:10.1109/tse.1983.235114.

[2] Buhrer, H. K. Software development. *ACM SIGSOFT Software Engineering Notes.* 28(2), 5. http://doi.org/10.1145/638750.638777.

[3] Jacobson, I., and Seidewitz, E. A new software engineering. *Communications of the ACM.* 57(12), 49–54. http://doi.org/10.1145/2677034.

[4] Poonam Narang, Pooja Mittal, Preeti Gulia and Balkrishan, "*Insights into DevOps automation tools employed at different stages of software development*", in Book

Computational Intelligence in Software Modeling Computational Intelligence in Software Modeling, edited by Vishal Jain, Jyotir Moy Chatterjee, Ankita Bansal, UtkuKose and Abha Jain, Berlin, Boston: De Gruyter, 2022, pp. 93-106. https://doi.org/10.1515/9783110709247-007

[5] Henry Edison, Xiaofeng Wang and Kieran Conboy. Comparing Methods for Large-Scale Agile Software Development: A Systematic Literature Review. *IEEE Transactions on Software Engineering.* SE-48(8), 2709–2731.http://doi.org/10.1109/TSE.2021.3069039.

[6] Poonam Narang and Pooja Mittal. Software Development Methodologies: Trending from Traditional to DOSE – An Empirical Study. *Proc of IEEE Delhi Section International Conference on Electrical, Electronics and Computer Engineering (DELCON-2022).* Available online at: https://ieeexplore.ieee.org/document/9753613

[7] Luca Traini. Exploring Performance Assurance Practices and Challenges in Agile Software Development: An Ethnographic Study. *Empirical Software engineering* (2022) 27:74. http://doi.org/10.1007/s10664-021-10069-3.

[8] Eliezio Soares, Gustavo Sizilio, Jadson Santos, Daniel Alencar da Costa and Uira Kulesza. The effects of continuous integration on software development: a systematic literature review. *Empirical Software Engineering* (2022) 27:78. doi:10.1007/s10664-021-10114-1

[9] Poonam and Pooja Mittal. DevOps- Bringing efficiency in delivering Software Product: A Review. *Proc of National Conference on Future Innovations in Computing Technologies & Machine Learning (FICTML-17).* Maharshi Dayanand University, Rohtak (Haryana), November 21, 2017, ISBN 978-93-80544-31-1

[10] Dilara Atesogullari and Alok Mishra (2020). Automation Testing Tools: A Comparative View. *International Journal on Information Technologies & Security.* 4(12), 2020.

[11] Elis Pelivani, Adrian Besimi and Betim Cico. An Empirical Study of User Interface Testing Tools. *International Journal on Information Technologies & Security.* 1(14), 2022.

[12] Dhis Elhaq Rzig, Foyzul Hassan and Marouane Kessentini. *In book Information and Software Technology.* Vol 152, December 2022, 107037. http://doi.org/10.1016/j.infsof.2022.107037.

[13] Poonam Narang and Pooja Mittal. Hybrid Model for Software Development: an Integrated Comparison of DevOps Automation Tools. *Indonesian Journal of Electrical Engineering and Computer Science*. Vol 27, No 1, July 2022, 456~465. http://doi.org/10.11591/ijeecs.v27.i1.pp456-465.

[14] Poonam Narang and Pooja Mittal. Implementation of DevOps Hybrid Model for Project Management and Deployment using Jenkins with Plugins. International Journal of Computer Systems and Network Security. Vol 22, No 8, Aug 2022, 249-259. https://doi.org/10.22937/IJCSNS.2022.22.8.31

[15] Poonam Narang and Pooja Mittal. Performance Assessment of Traditional Software Development Methodologies and DevOps Automation Culture. *Engineering Technology & Applied Science Research (ETASR)*, Accepted for Publication.

[16] Winston W. Royce. *Managing the development of large software systems.* Available at: https://en.wikipedia.org/wiki/Winston_W._Royce (visited on 14.09.2022).

[17] Georgios Papadopoulos. Moving from Traditional to Agile Software Development Methodologies Also on Large, Distributed Projects. *Elsevier Procedia – Social and*

Behavioral Sciences, Vol 175, Feb 2015, pp 455-463. https://doi.org/10.1016/j.sbspro.2015.01.1223

[18] Omer Uludag, Abheeshta Putta, Maria Paassivaara and Florian Matthes. Evolution of the Agile Scaling Frameworks. *International Conference on Agile Software Development, XP 2021*. LNBIP, Volume 419. https://doi.org/10.1007/978-3-030-78098-2_8.

[19] Torgeir Dingsoyr, Nils Brede Moe, Tor Erlend Faegri and Eva Amdahl Seim. Exploring Software Development at the very large-scale: a revelatory case study and research agenda for agile method adaptation. *Empirical Software Engineering* 23. 490- 520 (2018). https://doi.org/10.1007/s10664-017-9524-2.

[20] Ashish Agrawal, Mohd, Auranzeb Atiq and L.S. Maurya. A Current Study on the Limitation of Agile Methods in Industry Using Secure Google Forms. *Elsevier Procedia Computer Science.* Vol 78, 2016, pp 291-297. https://doi.org/10.1016/j.procs.2016.02.056.

[21] Marta Gomes, Ruben Pereira, Miguel Silva, Jose Braga de Vasconcelos and Alvaro Rocha. KPIs for Evaluation of DevOps Teams. In book Information Systems and Technologies. LNNS Series, Vol 470, pp 142-156. https://doi.org/10.1007/978-3-031-04829-6_13.

[22] Alok Mishra and Ziadoon Otaiwi. DevOps and software quality: a systematic mapping. *Elsevier Computer Science Review.* Vol 38, Nov 2020, 100308. https://doi.org/10.1016/j.cosrev.2020.100308.

[23] Debois P. Agile infrastructure and operations: how infra-gile are you? *Proc of the Agile 2008 Conference.* IEEE, Toronto, ON,Canada, ISBN: 978-0-7695-3321-6. https://doi.org/10.1109/Agile.2008.42.

[24] Khan AA and Shameem M. Multicriteria decision-making taxonomy for DevOps challenging factors using analytical hierarchy process. *Journal of Software Evolution and Process*. 2020, 32(10), pp 11-13, https://doi.org/e2263.10.1002/smr.2263.

[25] Leite L, Rocha C, Kon F, Milojicic D and Meirelles P. A survey of DevOps concepts and challenges. *ACM Computing Surveys*. 2019, 52(6), pp 1-35. https://doi.org/10.1145/3359981.

[26] Trihinas D, Tryfonos A, Dikaiakos MD and Pallis G. DevOps as a service: pushing the boundaries of microservice adoption. *IEEE Internet Computing. 2*2(3), pp 65-71.

Information about the authors:

Poonam Narang –Poonam Narang, Research Scholar is pursuing PhD from the Department of Computer Science and Applications, Maharshi Dayanand University, Rohtak, Haryana under the supervision of Respected Dr. Pooja Mittal (Research Guide and Second Author). Author's Qualification is M.Phil. (CS), MCA. Her work is published in many National and International Conferences including Elsevier, IEEE and many renowned Journals including Scopus and ESCI indexed. She can be contacted at email: poonam.mehta20@gmail.com

Pooja Mittal – Dr. Pooja Mittal obtained her Ph.D. degree from MDU. Her area of research and specialization include Data Mining, Data Warehousing, and Computer Science. She had published more than 50 research papers in renowned International and National Journals and attended more than 30 Conferences. Currently she is working as Assistant Professor in the DCSA, MDU, Rohtak (Haryana). She can be contacted at email: mpoojamdu@gmail.com

Manuscript received on 14 September 2022
Revised manuscript re-submitted on 20 October 2022