The Development of a Simulation Model for Assessing the CI/CD Pipeline Quality in the Development of Information Systems Based on a Multi-Agent Approach

Denis Medvedev^{1,*} and *Konstantin Aksyonov*¹

¹Engineering School of Information Technologies, Telecommunications and Control Systems, Ural Federal University, Ekaterinburg, Russia

Abstract. The need for the development of fast and efficient software requires the automation during the build, test and release stages. Thus, a methodology of the continuous integration and continuous delivery (CI/CD) emerges which using a continuous integration server which makes easier automatical build, test, and package of the new feature settings. Poor understanding and execution of CI/CD operations can slow down and even stop the pace of a software project. Many bottlenecks in the CI/CD pipeline can arise from misconfiguration, in other words, an inadequate level of automation, inappropriate agent characteristics, and a suboptimal queuing strategy. On the other hand, the finding a way to analyze and improve the settings of the CI/CD pipeline brings great management and economic benefits, since an optimal configuration ultimately implies high efficiency. In this regard, this study attempts to develop a simulation model that will provide a testing environment for a variety of factors that affect CI/ CD performance. The present study, therefore, aims to provide a pipeline simulation model based on the logic of a queuing system, which is based on a multi-agent approach and an experimental environment that allows you to test different settings and scenarios.

1 Introduction

The study is devoted to solving the problem of development of a simulation model for the CI/CD concept in the field of creating information systems (IS) based on a multi-agent approach.

2 Pipeline CI/CD

CI/CD is a technology for automating the build, testing, and delivery of the new modules of development project to the parties concerned. The Continuous Integration (CI) and Continuous Delivery (CD) are two aspects of the CI/CD technology, as well as a set of

*Corresponding author: medvedev.denis@urfu.ru

principles and practices that allow to speed up the process of building, testing, and deploying an application.

The Continuous Integration is a methodology of development and set of practices in which developers regularly merge changes to the application code in a central repository, after which the application is automatically built, tested, and launched. The main goal of the continuous integration is to automate the process of building and testing of the application in order to quickly find and fix errors, improve quality, and reduce the time spent on checking and releasing new software updates.

The Continuous Delivery is a methodology of development that extends the practice of continuous integration by ensuring that all code changes after the building stage are deployed in a test and/or production environment. With the right implementation, the developers will always have a built-in software unit ready for delivery, which has passed a standardized testing procedure [1].

Fig. 1. Example of CI/CD pipeline

Together, these practices form the CI/CD pipeline, which is shown in Figure 1. All the stages presented in the diagram, except for fixing source code changes, are performed on a special server or group of servers that act as agents, and a special dedicated continuous integration server is engaged in the orchestration of the entire process. After changing the source code of the application, the developer sends it to the central repository and the continuous integration server automatically selects the first free agent from the pool and sends the build task to it. When the agent receives a task, it builds the application and notifies the continuous integration server of the build result. If the build was successful, the continuous integration server again selects the first free agent from the pool and sends the task to it. After completing the tests, the continuous integration server provides a report on passed or failed tests and deliveries the application to the required environment.

3 Tasks description

Due to the fact that the number of agents is always finite, there is a situation when, with a large flow of source code changes, we can see a queue of tasks for building, testing and deploying the application. In turn, this leads to a longer product lifecycle and slower development, which can negatively affect the commercial performance of the application. Queuing theory offers several ways of the effectively use it to avoid such queues. The main task of queuing theory is to establish an optimal dependency between the number of service channels and their performance. The optimal dependency involves minimizing the costs of service channels and losses associated with downtime of the technological equipment in the service queue and with downtime of the service channels [2].

In this case, each agent in the CI/CD pipeline is a service node, and the entire CI/CD pipeline acts as a queuing multi-channel system (QMS). The sources of requests are developers who submit changes to the application source code to the central repository.

Due to the fact that each service node has a fixed cost, the task is to find the indicators for the queuing system and establish an optimal dependency between the number of service channels and their performance for a certain number of events per period.

One of the methods for calculating the indicators that ensure the effectiveness of the queuing system is the method of simulation modeling. It was proposed a simulation model of the CI/CD pipeline with sequential execution of each stage, which is presented in Fig. 2 and the pipeline model with parallel execution of the testing stage, which is presented in Fig. 3 to solve this problem. The design of various types of software based on simulation modeling is devoted to the following studies. [3-7]. Various aspects of software development are devoted to the following studies [8-15].

Agents have a set of the following characteristics:

- maintainable task types – it defines what types of tasks can be performed on a particular agent;
- service cost per month – the cost for agent operating;
- accessibility – if the agent can accept requests;
- probability of the agent failure – it describes the probability of the agent fail and requiring the service.

Fig. 2. CI/CD pipeline in simulation model

The input flow is tasks from the continuous integration server that have the following characteristics:

- type – it defines the type of task (application build, unit testing, integration testing, UI testing, manual testing, validation, deployment to the required environment);
- branch in the central repository - to download the necessary branch with the source code during the task execution;
- the duration of the task on the agent;
- error probabilities – it defines the probability of making errors during the execution of the task;
- error severity – it describes the impact of the error on the task execution.

Fig. 3. CI/CD pipeline with parallel test phase

The input flow of requests is random, since the time between the two requests is a random amount and depends on the submission of changes to the application source code to the central repository by the developers and on the completion of previous tasks on other agents. The intensity of the flow λ is the frequency of the occurrence of events entering the queuing multi-channel system per unit of time. The number of input flows entering the queuing multi-channel system is equal to the number of agents. Depending on the type of tasks, it is possible to perform them in parallel on free agents. These tasks include testing tasks.

The Queue is the place where the received requests are waiting for the start of service. In this case, the queue has an unlimited length. In addition, the waiting time in the queue is also unlimited. Queue maintenance discipline is: first in, first out.

In the test environment taken for development of the simulation model, the following values of the starting parameters were determined:

- $\lambda = 20 \text{ requests/day}$;
- $K = 20 - \text{number of channels in the system}$;
- $T_{\text{mean build time}} = 28 \pm 7 \text{ minute}$;
- $T_{\text{mean unit-tests time}} = 6 \pm 2 \text{ minute}$;
- $T_{\text{mean integration tests time}} = 183 \pm 12 \text{ minute}$;
- $T_{\text{mean UI tests time}} = 47 \pm 9 \text{ minute}$;
- $T_{\text{mean deployment}} = 52 \pm 6 \text{ minute}$.

To assess the quality of the received QS, the following indicators were selected:

- throughput of the system;
- the probability of each channel being busy and all together;
- mean busy time of each channel;
- the probability of all channels being busy;
- mean number of busy channels;
- probability of downtime of each channel;

- probability of downtime of the entire system;
- the mean number of requests in queue;
- mean waiting time for a request in queue;
- mean of service time;
- mean time spent by a request in the system.

4 Estimate and conclusion

The current study attempts to model the CI/CD pipeline in a simulation system using a multi-agent approach in order to establish an optimal dependency between the number of service channels and their performance at a certain number of events per period. The research process will include building and testing of the model. Thanks to this model, it will be possible to calculate the amount of resources required to work on a project more efficiently and take into account many parameters.

To sum up, we can say that the final product of the study has the following applications and advantages:

- Low cost: the research on the impact of the various parameters and attributes of the CI/CD pipeline is relatively cheap due to simulation modeling. However, conducting such a study in real life is quite expensive and time-consuming;
- through simulation, you can not only test different scenarios, but also track the causes of errors in advance by examining the output data. This may provide insight into some issues that may have important managerial and economic implications.

The work was supported by Act 211 Government of the Russian Federation, contract no. 02.A03.21.0006.

References

1. Amazon Web Services. "What is DevOps?" <https://aws.amazon.com/devops/what-is-devops/> (accessed Jan. 20, 2021).
2. A. A. Ivanov, "Simulation of automated production facilities based on queuing theory," (in Russian), Math. Proceedings of NSTU im. R. E. Alekseeva, 2013.
3. Sysoletin E., Aksyonov K., Nisskhen X., Aksyonova O. Development of components of multi-agent CASE-system for describing the logic of behavior of mobile robots. UKSim-AMSS 11th European Modelling Symposium on Mathematical Modelling and Computer Simulation. Manchester, England, 20 - 22 November 2017. Pages 3-8. DOI 10.1109/EMS.2017.12. WOS:000434813800001.
4. Medvedev S.N., Aksyonov K.A., Aksyonova O.P. Application of a decision support system in an industrial enterprise. IOP Conf. Series: Materials Science and Engineering. Vol. 709 (2020) 044026. doi:10.1088/1757-899X/709/4/044026 <https://iopscience.iop.org/article/10.1088/1757-899X/709/4/044026/pdf> International Conference on Modern Trends in Manufacturing Technologies and Equipment 2019, ICMTME 2019; Sevastopol; Russian Federation; 9-13 September 2019
5. A. Kruglov. Automated Monitoring and Control System for Forestry Enterprises. 5th International Young Scientists Conference on Information Technologies, Telecommunications and Control Systems, ITTCS 2018; Yekaterinburg; Russian Federation; 6-8 December 2018. Code 144071. CEUR Workshop Proceedings Volume 2298, 2018, P. 7. <http://ceur-ws.org/Vol-2298/paper23.pdf>

6. R. Drezewski. A model of co-evolution in multi-agent system, In: V. Marik, J. Muller & M. Pechoucek, (Ed.), Multi-Agent Systems and Applications III, Vol. 2691 of LNCS, Springer-Verlag, Berlin, Heidelberg, pp. 314-323.
7. J.P Muller & M.Pischel. 1993. The Agent Architecture InteRRaP: Concept and Application, German Research Center for Artificial Intelligence (DFKI). 1993.
8. K. Cetnarowicz, Kisiel-Dorohinicki, M. & E. Nawarecki. The application of evolution process in multi-agent world (MAW) to the prediction system, In: M. Tokoro, (Ed.), [Proc. of the 2nd Int. Conf. on Multi-Agent Systems (ICMAS 96), AAAI Press]
9. Andrey Tarasiev, Xenia Nisskhen, Margarita Filippova, Elena Smoliy. Application of the Modified Irwin Method for Development of the Detecting DDOS-Attacks System. 5th International Young Scientists Conference on Information Technologies, Telecommunications and Control Systems, ITTCS 2018; Yekaterinburg; Russian Federation; 6-8 December 2018. Code 144071. CEUR Workshop Proceedings Volume 2298, 2018, p. 7. <http://ceur-ws.org/Vol-2298/paper9.pdf>
10. Zhuravlev A.V., Aksyonov K.A. and Knyazev R.O. Methodology for building the optimal test bench of software and hardware complexes of control systems. Journal of Physics: Conference Series, Volume 1694 (2020) 012003, Information Technology, Telecommunications and Control Systems (ITTCS), 2020 17-18 December 2020 Innopolis, Russia. <https://iopscience.iop.org/article/10.1088/1742-6596/1694/1/012003> doi:10.1088/1742-6596/1694/1/012003
11. Shokhista Ergasheva, Artem Kruglov. Software Development Life Cycle early phases and quality metrics: A Systematic Literature Review. Journal of Physics: Conference Series, Volume 1694, Information Technology, Telecommunications and Control Systems (ITTCS), 2020 17-18 December 2020 Innopolis, Russia. doi:10.1088/1742-6596/1694/1/012007
12. Swati Megha, Hamza Salem, Enes Ayan, Manuel Mazzara. A Survey of Blockchain Solutions for Autonomous Vehicles Ecosystems. Journal of Physics: Conference Series, Volume 1694 (2020) 012024, Information Technology, Telecommunications and Control Systems (ITTCS), 2020 17-18 December 2020 Innopolis, Russia. doi:10.1088/1742-6596/1694/1/012024
13. Manuel Rodriguez, Luiz Jonatã Pires de Araraju and Manuel Mazzara. Good practices for the adoption of DataOps in the software industry. Journal of Physics: Conference Series, Volume 1694 (2020) 012032, Information Technology, Telecommunications and Control Systems (ITTCS), 2020 17-18 December 2020 Innopolis, Russia. doi:10.1088/1742-6596/1694/1/012032
14. K Fedoseev, N Askarbekuly, A E Uzbekova, M Mazzara. Application of Data-Oriented Design in Game Development. Journal of Physics: Conference Series, Volume 1694 (2020) 012035, Information Technology, Telecommunications and Control Systems (ITTCS), 2020 17-18 December 2020 Innopolis, Russia. doi:10.1088/1742-6596/1694/1/012035
15. Giancarlo Guizzardi, Gerd Wagner Tutorial: Conceptual simulation modeling with Onto-UML [Text] , (WSC 2012), December 9-12, 2012, Berlin, Germany.[Proceedings of the 2012 Winter Simulation Conference]