Jakob Maležič
Cesta v log 18, 1330 Kočevje, Slovenija
Študijski program: Računalništvo in informatika, MAG
Vpisna številka: 63170191

Komisija za študijske zadeve
Univerza v Ljubljani, Fakulteta za računalništvo in informatiko
Večna pot 113, 1000 Ljubljana

Vloga za prijavo teme magistrskega dela
Kandidat: Jakob Maležič

Jakob Maležič, študent magistrskega programa na Fakulteti za računalništvo in informatiko, zaprošam Komisijo za študijske zadeve, da odobri predloženo temo magistrskega dela z naslovom:

Slovenski: **Neprekinjena integracija in dostava poslovno kritičnih aplikacij**
Angleški: **Continuous integration and delivery for business critical applications**

Tema je bila že potrjena lani in je ponovno vložena: ***DA***

Izjavljam, da so spodaj navedeni mentorji predlog teme pregledali in odobrili ter da se z oddajo predloga strinjajo.

Magistrsko delo nameravam pisati v slovenščini.

Za mentorja/mentorico predlagam:

Ime in priimek, naziv: Nejc Ilc, dr. doc.
Ustanova: Fakulteta za Računalništvo in Informatiko
Elektronski naslov: nejc.ilc@fri.uni-lj.si

Za somentorja/somentorico predlagam:

Ime in priimek, naziv: Tadej Justin, dr.
Ustanova: Medius
Elektronski naslov: tadej.justin@medius.si

V Ljubljani, 1. december 2022.

PREDLOG TEME MAGISTRSKEGA DELA

1 Področje magistrskega dela

slovensko: razvijalci in operacije
angleško: developers and operations

2 Ključne besede

slovensko: neprekinjena integracija, neprekinjena dostava, poslovno kritične aplikacije
angleško: continuous integration, continuous deployment, business critical applications

3 Opis teme magistrskega dela

Pretekle potrditve predložene teme:
Predložena tema je bila oddana in potrjena v preteklih letih ter se od lanske ne razlikuje.

3.1 Uvod in opis problema

Pri razvoju večje programske opreme ali aplikacije je programje navadno razdeljeno na več zaključenih enot, ki jih različne razvojne skupine ali razvojna podjetja neodvisno razvijajo. Za slednje je pred delitvijo nalog vnaprej izbrano in specificirano tudi produkcijsko okolje. Na ta način si lahko razvojne skupine, za preverjanje realizacije izvedenega dela, pripravijo razvojno okolje, ki ima čim bolj podobno infrastrukturo, kot je na voljo v produkcijskem okolju. Samo tako so posamezne skupine lahko dovolj samozavestne, da bo njihovo programje pravilno delovalo tudi v produkcijskem okolju.

Poseben primer produkcijskega okolja predstavlja zaprto produkcijsko okolje naročnika, ki se velikokrat pojavi pri poslovno kritičnih aplikacijah. Poslovno kritične aplikacije so tiste aplikacije, ki so ključne za delovanje poslovnega procesa podjetja [1, 2]. V takšnem primeru je zaradi varnosti produkcijsko okolje pogosto razvijalcem nedostopno, kot tudi celotno omrežje naročnika. S tem je dostava programske kode v omrežje naročnika s strani razvijalcev nemogoča. Za takšen primer razvijanja programske opreme je potrebno celoten proces neprekinjene dostave prilagoditi in upoštevati morebitne dodatne zaplete pri izdaji programske opreme v produkcijsko okolje.

V magistrski nalogi bomo predstavili strategije in tehnologije, ki nam omogočajo postavitev takšnega sistema. Slednje bomo opisali na realnih projektih in pripravili orodja za izvajanje integracijskega ter dostavnega procesa. Bolj podrobno bomo opisali tudi proces neprekinjene integracije in dostave v okolje, kjer nimamo dostopa do nobenega izmed programskih okolij in nimamo možnosti odriva programske kode v naročnikovo omrežje.

3.2 Pregled sorodnih del

Zanimanje in motivacijo za vpeljevanje strategij razvoja in operacij smo našli v [3], kjer so predstavljene tehnike vpeljevanja strategij v realnih delovnih okoljih. Avtor daje velik poudarek motivaciji in razlogom zakaj je vpeljava takšnih strategij potrebna in kako ugotoviti, katera orodja zares potrebujemo. Predstavljen je tudi psihološki vidik uporabe takšnih orodji na zaposlene.

V [4] so opisani posamezni koraki neprekinjene integracije in dostave ter orodja, ki se v posameznih korakih uporabljajo. Opisan je namen posameznih orodji in kako pripomorejo k bolj učinkovitemu delovanju integracijskega procesa.

V [5] je podrobno opisana neprekinjena dostava po posameznih korakih v cevovodu na primeru velikega podjetja. V prispevku so opisane prednosti in izzivi vzpostavitve takšnega sistema.

V [6] so predstavljene raziskave na področju oblačnega računalništva in uporaba principov razvoja ter operacij na oblačni programski rešitvi. Na osnovi tega, so opisane dve arhitekturni zasnovi. V rezultatih se izkaže, da je proces namestitve nove programske opreme veliko lažji in hitrejši, če ga avtomatiziramo in upoštevamo principe razvoja in operacij. Opiše tudi slabosti in prednosti tehnologije Kubernetes, ki jo bomo pri implementaciji tudi sami uporabljali.

3.3 Predvideni prispevki magistrske naloge

V delu bomo predstavili izzive, prednosti in rešitve razvoja poslovno kritičnih aplikacij z uporabo neprekinjene integracije in dostave programske opreme. Opisali bomo različne tehnologije, ki so del tega procesa in njihovo uporabo predstavili na realnem primeru. Za potrebe delovanja takšnega cevovoda bomo razvili splošna orodja, ki nam omogočajo uporabo pripravljenega cevovoda na projektih s poljubno arhitekturo. Vsa orodja, ki jih bomo uporabljali, bodo dostopna zgolj znotraj lokalnega omrežja. S tem bomo zagotovili visoko stopnjo varnosti, ki je pomembna pri poslovno kritičnih aplikacijah. Največji prednosti razvitega sistema bosta možnost prilagoditve sistema za skoraj poljuben projekt in možnost izvajanja celotnega procesa integracije v naročnikovem okolju, neodvisno od

podjetja, ki je aplikacijo razvilo.

3.4 Metodologija

Pri pripravi cevovoda za neprekinjeno integracijo in neprekinjeno dostavo bomo uporabljali različne repozitorije programske opreme: Nexus, Git in register Docker. Za upravljanje s cevovodom in organizacijo dela bomo uporabljali platformo GitLab, delovanje vsebovalnikov Docker bomo upravljali s sistemom Kubernetes, konfiguracijo posameznih aplikacij pa s konfiguracijskim strežnikom Consul. Vsa orodja bomo medseboj povezali z uporabo skriptov napisanih v jeziku Bash. Izdelan cevovod in pripadajoča orodja bomo predstavili na realnem projektu in opisali njegove prednosti in slabosti.

3.5 Literatura in viri

[1] M. Hinchey, L. Coyle, Evolving Critical Systems: A Research Agenda for Computer-Based Systems, in: 2010 17th IEEE International Conference and Workshops on Engineering of Computer Based Systems, IEEE, 2010, pp. 430–435.

[2] J. Snygg, A. Smedberg, G. Juell-Skielse, Problem management of business-critical systems in literature: a review and viable systems analysis, International Journal of Business Continuity and Risk Management 6 (2016) 238.

[3] J. Smith, Operations anti-patterns, DevOps Solutions, Manning Publications, 2020.

[4] C. Ebert, G. Gallardo, J. Hernantes, N. Serrano, DevOps, IEEE Software 33 (3) (2016) 94–100.

[5] L. Chen, Continuous delivery, in: Proceedings of the International Workshop on Continuous Software Evolution and Delivery, ACM, 2016, pp. 84–84.

[6] P. Kočevar, Analysis and implementation of cloud-based architectures enabling software product quality (2019).