Zrelost cevovodov neprekinjene dostave informacijskih rešitev

Luka Četina, Luka Pavlič

Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko, Maribor, Slovenija
luka.cetina@um.si, luka.pavlic@um.si

DevOps se je uveljavil kot pomemben pristop k sodobnemu razvoju informacijskih rešitev, saj povezuje razvojne in operativne vidike ter organizacijam omogoča hitrejši, zanesljivejši razvoj. Ker DevOps ne predstavlja enotne metode, organizacije prakse uvajajo postopoma in različno, težko pa ocenijo, kako zreli so njihovi procesi in katere izboljšave so najbolj nujne. Obstoječi modeli zrelosti DevOps pri tem pomagajo, a so praviloma usmerjeni na organizacijsko raven, kulturo in procese, ocenjevanje pa temelji na samooceni s splošno opisanimi kriteriji, kar otežuje primerljivost ocen med ekipami. Manj pozornosti namenjajo konkretni tehnični izvedbi avtomatizacije – cevovodom neprekinjene dostave, prek katerih se velik del DevOps praks dejansko izvaja. Prispevek predstavlja preliminarno zasnovo modela za ocenjevanje zrelosti CI/CD cevovodov, ki temelji na kriterijih, preverljivih neposredno v konfiguraciji cevovoda. Obrazec zajema šest razsežnosti (avtomatizacija gradnje in integracije, testiranje in analiza kode, avtomatizacija izdaje in namestitve, infrastruktura kot koda, spremljanje izvajanja cevovoda ter varnost dobavne verige), vsaka s petimi kriteriji, ocenjenimi s 0 do 2 točkama. Skupna ocena (0–60 točk) se preslika v pet ravni zrelosti, razčlenitev po razsežnostih pa pokaže, kje je cevovod dodelan in kje zaostaja. Predlagani model predstavlja dopolnilo obstoječim modelom zrelosti DevOps, saj omogoča sistematično, ponovljivo in potencialno avtomatizirano ocenjevanje tehnične zrelosti posameznega CI/CD cevovoda.

Ključne besede:
CI/CD cevovodi
zrelost cevovodov
DevOps
model zrelosti
neprekinjena dostava

Univerzitetna založba Univerze v Mariboru

ISBN 978-961-299-188-3
DOI https://doi.org/10.18690/um.feri.7.2026.8

1 Uvod

Prakse neprekinjene integracije, dostave in namestitve (angl. continuous integration, continuous delivery, continuous deployment – CI/CD) so danes med temeljnimi gradniki sodobnega razvoja informacijskih rešitev. Neprekinjena integracija zahteva pogosto in avtomatizirano združevanje kode in tako zmanjšuje tveganje integracijskih napak, neprekinjena dostava pa ta koncept nadgradi z avtomatiziranim testiranjem in preverjanjem kakovosti, tako da je aplikacija vedno pripravljena na produkcijsko namestitev [1]. Neprekinjena namestitev gre še korak dlje in produkcijsko namestitev izvede samodejno, brez ročnega posredovanja [1]. Te prakse se izvajajo prek CI/CD cevovodov, ki avtomatizirajo aktivnosti, kot so gradnja, testiranje, analiza kode in namestitev [2].

CI/CD cevovodi so pomemben del metodologije DevOps, saj povezujejo razvojne in operativne vidike ter organizacijam omogočajo hitrejši, zanesljivejši in manj naporen razvoj [3]. Avtomatizacija teh aktivnosti pospešuje izdajne cikle, izboljšuje kakovost programske opreme in povečuje produktivnost razvijalcev [4]. Kljub tem koristim pa vzpostavitev in razvoj cevovodov v praksi neredko ostajata precej nenačrtovana, saj se cevovod pogosto razvija glede na trenutne potrebe, namesto da bi sledil strateškemu načrtu [5].

Da bi lahko razvoj cevovoda načrtovali in ga postopoma izboljševali, je treba najprej ugotoviti, v kakšnem stanju cevovod dejansko je [6]. Vendar za ocenjevanje njegove zrelosti ni uveljavljenega enotnega pristopa, saj ni jasno opredeljeno, katere značilnosti mora cevovod izpolnjevati, da bi ga lahko šteli za zrelega oziroma skladnega z načeli neprekinjene dostave [7]. Obstoječi modeli zrelosti DevOps sicer ponujajo okvir za ocenjevanje uvajanja DevOps praks, vendar praviloma obravnavajo organizacijo ali razvojno ekipo kot celoto. Njihovo ocenjevanje temelji predvsem na intervjujih ali samoocenjevalnih vprašalnikih [8], zato so primerni za presojo organizacijskih procesov, kulture in sodelovanja, bistveno manj pa za objektivno oceno tehnične izvedbe posameznega CI/CD cevovoda.

Ta prispevek predstavlja preliminarno zasnovo modela za ocenjevanje zrelosti CI/CD cevovodov, ki temelji na kriterijih, razvidnih neposredno iz konfiguracije cevovoda. Za razliko od obstoječih modelov zrelosti DevOps, ki ocenjujejo predvsem organizacijsko raven, predlagani pristop omogoča ocenjevanje tehnične zrelosti posameznega cevovoda brez uporabe intervjujev ali samoocenjevalnih vprašalnikov. Takšna ocena ekipam omogoča prepoznavanje konkretnih tehničnih vrzeli, spremljanje napredka skozi čas ter primerjavo cevovodov na enotni osnovi. V prispevku predstavimo razsežnosti predlaganega modela, način ocenjevanja in točkovanja ter njegovo uporabo na primeru hipotetičnega cevovoda. Ker gre za konceptualno zasnovo modela, predstavljeni kriteriji in pragovi še niso empirično validirani ter predstavljajo izhodišče za nadaljnji razvoj.

2 Ozadje in sorodna dela

2.1. CI/CD cevovodi kot koda

CI/CD cevovod je zaporedje avtomatiziranih korakov, ki se sprožijo ob dogodkih v razvojnem procesu (npr. nalaganju kode ali odprtju zahteve za združitev) in izvedejo aktivnosti, kot so gradnja, testiranje, analiza kode, izdaja in namestitev [2], [9]. Ključna lastnost sodobnih cevovodov je, da so vse pogosteje definirani deklarativno, v datotekah, ki se hranijo skupaj s programsko kodo (npr. GitHub Actions, GitLab CI, Azure DevOps) [5]. Cevovod tako postane samostojen izdelek, ki ga je mogoče verzionirati, pregledovati in sistematično ocenjevati na podlagi njegove konfiguracije [10], [5], [11]. Ta predstavlja strukturiran, strojno berljiv vir podatkov o tem, kaj cevovod vključuje oziroma avtomatizira.

2.2. Modeli zrelosti DevOps in njihove omejitve za oceno cevovoda

Vprašanje zrelosti sicer v okviru DevOps ni novo. Vzporedno s CI/CD cevovodi se je razvila vrsta modelov zrelosti DevOps, ki organizacijam pomagajo oceniti, kako celovito so DevOps prakse vpeljane [7]. Med bolj znanimi sta model Humble in Farley, ki zrelost povezuje s stopnjo avtomatizacije konfiguracije izdaj in namestitev [12], ter na sistematičnem pregledu literature zasnovan model Teixeira idr. [13], ki DevOps zrelost opredelijo prek

štirih razsežnosti: ljudje, proces, tehnologija in kultura. Sethupathy [14] je nedavno predstavil okvir zrelostnega modela, ki združuje elemente vitke (ang. lean) in agilne zrelosti z DevOps prakso.

Pregled večjega števila takih modelov [8], [15] pokaže nekaj ponavljajočih se značilnosti: modeli so praviloma večdimenzionalni, zajemajo tehnične, procesne in kulturne vidike, imajo približno pet ravni zrelosti, ocenjevanje pa največkrat temelji na samooceni prek vprašalnikov ali intervjujev. Kriteriji za posamezne ravni so pogosto opisani splošno (npr. "standardiziran proces"), kar otežuje objektivno in ponovljivo ocenjevanje. Merljivi kazalniki, kot jih predlagata DORA in Accelerate (pogostost namestitev, čas do uveljavitve spremembe, čas okrevanja, delež neuspešnih sprememb) [16], [17], dajejo vpogled v učinkovitost dostave, a ne pojasnijo, katere avtomatizacijske zmožnosti cevovodu manjkajo. Namenjeni so predvsem merjenju rezultatov procesa razvoja programske opreme in ne neposredni oceni tehnične implementacije CI/CD cevovoda.

Podobne omejitve veljajo za pobude, usmerjene specifično v cevovode. Model zrelosti neprekinjene dostave podjetja AWS, prilagojen po zgledu CMM [18], ponuja okvirne stopnje zrelosti od osnovne avtomatizacije do optimizirane neprekinjene dostave, vendar brez konkretnih kriterijev za zaznavanje posameznih zmožnosti [19]. Podobno velja za varnostno usmerjeno pobudo SLSA, ki opredeljuje ravni zrelosti za sledljivost gradnje in celovitost dobavne verige, a se osredotoča le na varnost [20], ter za OpenSSF Scorecard, ki repozitorije samodejno ovrednoti glede na varnostne in vzdrževalne dobre prakse, brez opredeljenih razsežnosti zrelosti [21].

Tudi novejše sorodne raziskave večinoma ostajajo na organizacijski ravni: DevOps-RAF ocenjuje pripravljenost organizacije na DevOps prek agilnega razvoja, operativnega dela in kulture [22], model SAMAM naslavlja predvsem uvajanje ekstremnega programiranja [23], model ASPLA pa zrelost agilnih produktnih linij v avtomobilski industriji [24]. Akbar idr. [25] obravnavajo človeški dejavnik pri DevSecOps, Wang idr. [26] so pokazali povezavo med zrelostjo avtomatizacije testiranja in kakovostjo izdelka, a le za eno aktivnost cevovoda, Zarour idr. [27] pa so raziskali sprejemanje DevOps procesov, ne pa zrelosti tehnične izvedbe.

Skupni imenovalec navedenih del je torej, da zrelost obravnavajo na ravni organizacije, procesa ali posamezne prakse, ne pa na ravni CI/CD cevovoda, ki DevOps avtomatizacijo dejansko izvaja. Prav ta vrzel je izhodišče modela, predstavljenega v nadaljevanju.

3 Predlagani model zrelosti CI/CD cevovodov

Zrelost CI/CD cevovoda v tem prispevku opredeljujemo kot stopnjo, do katere cevovod dosledno, ponovljivo in preverljivo izvaja aktivnosti neprekinjene integracije, dostave in namestitve, usklajene z uveljavljenimi praksami DevOps. Za razliko od obstoječih modelov zrelosti DevOps [8], [12], [13], [14], predlagani model zrelost ocenjuje na ravni posameznega cevovoda, izključno na podlagi tega, kar je razpoznavno iz njegove konfiguracijske datoteke (npr. delovni tok GitHub Actions, datoteka GitLab CI .yml ali cevovod Azure DevOps) [5]. Model je zasnovan kot ocenjevalni obrazec in vsebuje nabor konkretnih, preverljivih kriterijev, razporejenih v razsežnosti, ki jih ocenjevalec preveri na dejanskem cevovodu in za vsakega dodeli točke. Vsota točk daje oceno zrelosti razsežnosti in na koncu skupno oceno zrelosti cevovoda.

Podobno kot novejši modeli zrelosti DevOps, ki ocenjevanje gradijo na konkretnih zmožnostih in točkovanju namesto zgolj opisnih ravni [14], tudi predlagani model kriterije formulira tako, da je odgovor nanje mogoče preveriti, ne le subjektivno oceniti. Razlika je v tem, da model ostaja izključno na ravni cevovoda: ne vključuje kriterijev, ki bi zahtevali vpogled v organizacijsko kulturo, sodelovanje ali kadrovske vidike, saj ti niso razvidni iz konfiguracije.

V nadaljevanju opišemo razsežnosti in njihove kriterije ter pristop ponazorimo na primeru hipotetičnega cevovoda.

3.1. Razsežnosti modela

Izbira razsežnosti izhaja iz dveh virov: pregleda obstoječih modelov zrelosti DevOps [7], [8], [12], [13], [14], [15], [18], [22], [23], [24], [25], [26] in omejitve, da je vsak kriterij mogoče preveriti neposredno iz konfiguracije cevovoda.

Pregled pokaže, da so med najpogosteje vključenimi razsežnostmi avtomatizacija (v vseh modelih), kultura, sodelovanje in proces (v skoraj vseh), spremljanje (angl. monitoring), arhitektura oziroma infrastruktura ter kakovost/testiranje, medtem ko je varnost kot samostojna razsežnost prisotna le v manjšini modelov [14]. To kaže, da modeli zrelosti DevOps varnost dobavne verige sistematično podcenjujejo, čeprav so orodja za njeno avtomatizirano preverjanje (npr. SLSA [20], OpenSSF Scorecard [21]) že razmeroma uveljavljena in, za razliko od marsikaterega drugega vidika DevOps prakse, neposredno razvidna iz konfiguracije cevovoda. Varnost dobavne verige zato v predlaganem obrazcu dobi status samostojne razsežnosti.

Kakovost programske kode (npr. statična analiza, upoštevanje standardov kodiranja) v predlaganem obrazcu ni samostojna razsežnost, čeprav se v pregledanih modelih pojavlja pogosteje kot varnost [15]. Namesto tega je zajeta kot del razsežnosti "Testiranje in analiza kode". Tak pristop – kakovost kode kot sestavni del širše razsežnosti kakovosti oziroma testiranja – je uveljavljen tudi v več pregledanih modelih [8], [15].

Na podlagi navedenega predlagamo šest razsežnosti obrazca:

- *Avtomatizacija gradnje in integracije*: ali cevovod samodejno preverja in združuje spremembe kode.
- *Testiranje in analiza kode*: ali cevovod samodejno preverja funkcionalno pravilnost, pokritost s testi in kakovost izvorne kode.
- *Avtomatizacija izdaje in namestitve*: ali cevovod samodejno pripravi in namesti izdelek.
- *Infrastruktura kot koda*: ali je infrastruktura, na katero se namešča izdelek, definirana in nadzorovana skozi cevovod.
- *Spremljanje izvajanja cevovoda*: ali je izvajanje cevovoda mogoče spremljati in ali se ob napakah samodejno obvešča.
- *Varnost dobavne verige*: ali cevovod samodejno preverja varnost odvisnosti, artefaktov in samega procesa gradnje.

3.2. Kriteriji in način ocenjevanja

Za vsako od šestih razsežnosti predvidevamo nabor konkretnih kriterijev, ki jih je mogoče preveriti neposredno v konfiguracijski datoteki cevovoda, brez potrebe po dodatnem razgovoru ali dostopu do organizacije. Vsak kriterij se oceni s tremi možnimi vrednostmi:

0 točk - kriterij v konfiguraciji ni zaznan (korak, sprožilec ali mehanizem ne obstaja);

1 točka – kriterij je zaznan delno (npr. korak obstaja, vendar ne za vse veje/okolja, ali je nastavljen zgolj informativno brez vpliva na izvajanje);

2 točki – kriterij je v celoti izpolnjen (korak je prisoten, se izvaja dosledno in dejansko vpliva na potek oz. izid cevovoda).

Ocena razsežnosti je vsota točk njenih petih kriterijev (0–10 točk), skupna ocena cevovoda pa vsota šestih razsežnosti (0–60 točk). Primeri kriterijev za razsežnost "Avtomatizacija gradnje in integracije" so prikazani v tabeli 1.

Tabela 1: Primer kriterijev za razsežnost "Avtomatizacija gradnje in integracije"

Št.	**Kriterij**
1	Cevovod se samodejno sproži ob potisu kode ali zahtevi za združitev, ne le ročno.
2	Neuspešna gradnja samodejno prepreči združitev v ciljno vejo.
3	Izdelan artefakt se shrani sledljivo (npr. z oznako različice).
4	Odvisnosti se namestijo samodejno, z zaklenjenimi/fiksiranimi različicami (lockfile).
5	Gradnja je predpomnjena oz. optimizirana (cache korakov, vzporedni jobi).

Za lažjo interpretacijo se točke preslikajo v eno od petih opisnih ravni. Najprej za vsako razsežnost posebej, šele nato se te sestavijo v skupno oceno, saj bi slednja sama lahko prikrila, da je cevovod na eni razsežnosti zelo dodelan, na drugi pa komaj vzpostavljen.

3.3. Ravni zrelosti

Skupno število točk (0–60) se za lažjo interpretacijo preslika v pet opisnih ravni zrelosti, ki so po zgledu večine pregledanih modelov zrelosti DevOps oblikovane naraščajoče, od povsem ročnega oziroma priložnostnega stanja do optimiziranega cevovoda [7], [8], [13]:

Raven 1 – začetna (0–12 točk): cevovod obstaja zgolj v osnovni obliki ali ga (še) ni; večina aktivnosti, vključno s pregledom kode, se izvaja ročno zunaj cevovoda.

Raven 2 – osnovna (13–24 točk): cevovod avtomatizira gradnjo in del testiranja, druge razsežnosti (namestitev, infrastruktura, varnost, spremljanje) so še večinoma ročne ali odsotne.

Raven 3 – definirana (25–36 točk): večina razsežnosti je delno avtomatizirana, vendar neenakomerno – npr. testiranje in gradnja sta dodelana, medtem ko varnost dobavne verige ali spremljanje izvajanja zaostajata.

Raven 4 – napredna (37–48 točk): cevovod dosledno avtomatizira večino aktivnosti po vseh razsežnostih, vključno z varnostnimi preverjanji in obveščanjem ob napakah; ročni koraki so izjema.

Raven 5 – optimizirana (49–60 točk): cevovod celovito avtomatizira aktivnosti po vseh razsežnostih, uporablja napredne prakse, kot so postopne izdaje, samodejna povrnitev na prejšnjo različico in sledljivo poreklo artefaktov, ter samodejno zaznava napake in nepravilnosti med izvajanjem.

Enak razpon ravni je smiselno uporabiti tudi na ravni posamezne razsežnosti (0–10 točk, pragovi sorazmerno nižji: 0–2, 3–4, 5–6, 7–8, 9–10), kar omogoča, da se poleg skupne ocene identificira tudi razsežnosti, kjer je cevovod razmeroma šibkejši – kot že omenjeno, je to za praktično uporabo modela pogosto bolj informativno kot sama skupna ocena. Tabela 2 za vsako od šestih razsežnosti opisuje, kaj naj bi jo opredeljevalo na posamezni od petih ravni.

Pri tem najvišja raven ni nujno cilj za vsak cevovod oziroma vsako organizacijo. Za majhen projekt ali zgodnjo fazo razvoja je lahko smiselno ciljati na 2. ali 3. raven, saj so stroški vzpostavitve naprednejših praks (npr. postopnih izdaj, izvoza metrik v namensko nadzorno ploščo) lahko nesorazmerni s tveganjem, ki ga cevovod dejansko nosi. Za cevovode, ki namestitve izvajajo v produkcijsko okolje z visoko izpostavljenostjo (npr. plačilni sistemi, javno dostopne storitve z veliko uporabniki), pa je pričakovati, da bi morali stremeti vsaj proti 4. ravni, predvsem na razsežnostih varnosti dobavne verige in spremljanja izvajanja, kjer so tveganja največja. Podobno razlikovanje ciljne ravni glede na kontekst organizacije je uveljavljeno tudi pri obstoječih modelih zrelosti DevOps [7], [8].

Tabela 2: Opis petih ravni zrelosti po posamezni razsežnosti obrazca

Razsežnost	Raven 1: Začetna	Raven 2: Osnovna	Raven 3: Definirana	Raven 4: Napredna	Raven 5: Optimizirana
Avtomatizacija gradnje in integracije	gradnja ročna, brez samodejnega sprožilca.	samodejna gradnja ob spremembi kode; neuspeh ne prepreči združitve, odvisnosti niso zaklenjene.	uspešna gradnja je pogoj za združitev (build gate); odvisnosti so zaklenjene na fiksne različice.	gradnja je pospešena s predpomnjenjem in vzporednim izvajanjem; gradijo se le spremenjeni deli.	podpora gradnji za več okolij/konfiguracij hkrati prek skupnih predlog (pipeline-as-code).
Testiranje in analiza kode	testi (če obstajajo) potekajo ročno; statične analize ni.	samodejni testi enot in osnovna statična analiza, oba le informativna.	neuspeh testa ustavi cevovod (test gate); tudi kršitve pravil statične analize blokirajo združitev.	dodani integracijski/sistemski testi, prag pokritosti kode; strožja pravila statične analize (npr. varnostna) so obvezujoča.	testiranje na matriki okolij ali dodatni tipi testov; trend kakovosti kode se spremlja skozi čas.
Avtomatizacija izdaje in namestitve	izdaja in namestitev sta ročni.	cevovod samodejno pripravi artefakt (npr. vsebnik), namestitev ostaja ročna.	namestitev v testno okolje je avtomatizirana, v produkcijo pa zahteva ročen postopek.	namestitev v produkcijo je avtomatizirana ali zahteva le eno potrditev; artefakti so sledljivi v registru.	napredne strategije izdaje s samodejnim rollbackom ob napaki.
Infrastruktura kot koda	infrastruktura je konfigurirana ročno.	infrastruktura je opisana kot koda (npr. Terraform), ločeno od cevovoda, brez preverjanja.	cevovod preveri veljavnost infrastrukturne kode (lint/plan) pred uveljavitvijo, ki je še delno ročna.	uveljavitev sprememb infrastrukture je v celoti avtomatizirana; konfiguracija in skrivnosti so ločene od kode.	infrastrukturna koda je upravljana po različicah in sledljiva (remote state) ter gre skozi enak pregled kot aplikacijska koda.
Spremljanje izvajanja cevovoda	o izidu izvajanja se ekipa seznani le naključno.	samodejno obvestilo ob neuspehu (e-pošta, Slack), dnevniki se ne arhivirajo posebej.	status izvajanja je viden zunaj cevovoda (značka); dnevniki in poročila se hranijo kot artefakt.	cevovod izvaža metrike delovanja (čas, uspešnost) v namensko nadzorno ploščo.	zaznava običanih/nenavadno dolgih izvajanj in samodejno opozarjanje na odstopanja.
Varnost dobavne verige	odvisnosti/artefakti se ne preverjajo, skrivnosti so lahko v kodi.	skrivnosti so upravljane prek namenskega mehanizma orodja CI/CD, preverjanja ranljivosti še ni.	korak za preverjanje znanih ranljivosti v odvisnostih (SCA orodje).	preverjajo se tudi ranljivosti gradbenih artefaktov/vsebnikov in izvor paketov.	artefakti imajo sledljivo poreklo (npr. SBOM, podpis, poročilo skladno z okvirom SLSA [20]).

3.4. Primer ocenjevanja

Predlagani obrazec za primer uporabimo na hipotetičnem, a tipičnem cevovodu manjše spletne aplikacije v orodju za CI/CD (npr. GitHub Actions):

"Cevovod se samodejno sproži ob vsakem potisu kode; uspešna gradnja je pogoj za združitev, odvisnosti so zaklenjene, predpomnjenja ali vzporednega izvajanja pa ne uporablja. Izvajajo se testi enot in preverjanje sloga kode, integracijski testi in pragovi pokritosti niso vključeni. Cevovod zgradi vsebnik in ga označi z različico ter ga

samodejno namesti v testno okolje; namestitev v produkcijo zahteva ročno potrditev, naprednejših strategij izdaje pa ne uporablja. Infrastruktura je opisana v Kubernetes manifestih, ki pred uveljavitvijo niso samodejno preverjeni; skrivnosti so upravljane prek vgrajenega mehanizma orodja za CI/CD. Ob napaki cevovod pošlje obvestilo v Slack, ne izvaža pa metrik o delovanju niti ne zaznava obtičanih izvajanj. Preverjanje ranljivosti odvisnosti, skeniranje artefaktov in beleženje porekla (SBOM) niso vključeni."

Na podlagi opisanega cevovoda posamezne razsežnosti ocenimo, kot prikazuje tabela 3.

Tabela 3: Ocena primera cevovoda po razsežnostih

Razsežnost	**Ocena (od 10)**
Avtomatizacija gradnje in integracije	8
Testiranje in analiza kode	4
Avtomatizacija izdaje in namestitve	7
Infrastruktura kot koda	4
Spremljanje izvajanja cevovoda	2
Varnost dobavne verige	2
Skupaj (od 60)	27

Skupna ocena cevovoda je tako 27 od 60 točk, kar ga uvršča v raven 3 – definirana. Ocena po razsežnostih (tabela 3) hkrati pokaže, kje je cevovod razmeroma dodelan (gradnja in integracija: 8/10) in kje precej zaostaja (spremljanje izvajanja in varnost dobavne verige: po 2/10). Taka razčlenitev je za ekipo uporabnejša od same skupne ocene, saj neposredno nakazuje, kam usmeriti naslednje izboljšave. V tem primeru najprej v uvedbo preverjanja ranljivosti odvisnosti in izvoz osnovnih metrik o delovanju cevovoda, saj gre za razmeroma enostavne posege z velikim vplivom na oceno.

3.5. Predviden način uporabe

Ideja predlaganega modela je, da ga lahko izpolni razvijalec ali vodja ekipe v razmeroma kratkem času, brez posebnega orodja, saj zadostuje vpogled v konfiguracijsko datoteko cevovoda in nastavitve orodja za CI/CD (npr. zaščita vej, register skrivnosti). Prepoznavanje kriterija je zasnovano kot preverjanje konkretnega, v konfiguraciji vidnega dejstva: npr. za kriterij "ali neuspešen test prepreči združitev kode" ocenjevalec preveri, ali je v nastavitvah zaščite veje vključeno zahtevano preverjanje stanja (angl. required status check) oziroma ali korak testiranja v konfiguraciji nima nastavljene možnosti za prezrtje napake (npr. allow_failure). Predviden postopek obsega tri korake: za vsak kriterij se preveri, ali je izpolnjen v celoti, delno ali sploh ne (0, 1 ali 2 točki); točke se seštejejo po razsežnostih in skupno, kar da profil zrelosti cevovoda; na podlagi profila ekipa prepozna razsežnosti z najnižjo oceno in jih prednostno naslovi, saj profil pove ne le, "kako zrel" je cevovod, temveč tudi, kje so vrzeli.

4 Praktične implikacije in omejitve

Predlagani model predstavlja praktično orodje za sistematično ocenjevanje tehnične zrelosti CI/CD cevovodov. Za razliko od obstoječih modelov zrelosti DevOps omogoča neposredno oceno tehničnih zmožnosti posameznega cevovoda na podlagi njegove konfiguracije. Razvijalcem in ekipam tako omogoča hitro prepoznavanje področij, kjer avtomatizacija še ni zadostno razvita, ter lažje načrtovanje nadaljnjih izboljšav.

Ker model podaja oceno po posameznih razsežnostih in ne zgolj skupne ocene, omogoča natančnejšo identifikacijo prednosti in pomanjkljivosti posameznega cevovoda. Tak pristop lahko organizacije uporabijo pri primerjavi različnih projektov, spremljanju razvoja cevovoda skozi čas ali kot podporo pri uvajanju dobrih praks DevOps. Obrazec lahko služi tudi kot kontrolni seznam pri načrtovanju novih cevovodov, saj razvijalcem pomaga sistematično preveriti, katere avtomatizacijske zmožnosti so že vključene in katere še manjkajo.

Pomembna prednost predlaganega pristopa je, da vsi kriteriji temeljijo na značilnostih, ki so neposredno preverljive iz konfiguracije cevovoda. Takšna zasnova zmanjšuje subjektivnost ocenjevanja, omogoča ponovljivost rezultatov med različnimi ocenjevalci ter odpira možnost delne ali popolne avtomatizacije ocenjevanja. V prihodnosti bi bilo mogoče razviti programsko orodje, ki bi iz konfiguracije cevovoda samodejno prepoznalo posamezne kriterije in izračunalo oceno zrelosti brez ročnega pregledovanja konfiguracije.

Predlagani model ima tudi nekaj omejitev. Ker temelji izključno na konfiguraciji cevovoda, ne zajema praks, ki se izvajajo zunaj njega. Poleg tega so predstavljene razsežnosti, kriteriji in pragovi v tej fazi še konceptualni ter niso bili empirično validirani. Nadaljnje delo bo zato usmerjeno v ekspertno presojo kriterijev, preverjanje njihove zanesljivosti med različnimi ocenjevalci ter uporabo modela na večjem številu javno dostopnih CI/CD cevovodov.

5 Zaključek

CI/CD cevovodi predstavljajo osrednji tehnični mehanizem izvajanja DevOps praks, vendar obstoječi modeli zrelosti njihovo ocenjevanje večinoma izvajajo na ravni organizacije oziroma razvojne ekipe. V prispevku smo predstavili konceptualno zasnovo modela za ocenjevanje tehnične zrelosti CI/CD cevovodov, ki temelji na preverljivih kriterijih, razvidnih neposredno iz konfiguracije cevovoda. Model vključuje šest razsežnosti, ki skupaj omogočajo sistematično oceno zrelosti ter prepoznavanje konkretnih področij, kjer je mogoče avtomatizacijo še izboljšati.

Predlagani pristop predstavlja dopolnilo obstoječim modelom zrelosti DevOps, saj se namesto organizacijskih dejavnikov osredotoča na tehnično implementacijo avtomatizacije v posameznem cevovodu. Ker temelji na jasno opredeljenih in preverljivih kriterijih, omogoča ponovljivo ocenjevanje različnih cevovodov ter odpira možnost njihove primerjave in spremljanja razvoja skozi čas. Hkrati takšna zasnova predstavlja osnovo za prihodnjo delno ali popolno avtomatizacijo ocenjevanja.

Ker je predstavljeni model še v konceptualni fazi razvoja, bo nadaljnje delo usmerjeno v ekspertno validacijo razsežnosti in kriterijev, preverjanje zanesljivosti ocenjevanja med različnimi ocenjevalci ter uporabo modela na večjem številu javno dostopnih CI/CD cevovodov. Rezultati teh analiz bodo podlaga za nadaljnjo izpopolnitev modela in njegovo empirično validacijo.

Literatura

[1] M. Shahin, M. Ali Babar, in L. Zhu, "Continuous Integration, Delivery and Deployment: A Systematic Review on Approaches, Tools, Challenges and Practices," IEEE Access, let. 5, str. 3909-3943, 2017, doi: 10.1109/ACCESS.2017.2685629.

[2] "What is a CI/CD pipeline?" Dostopno: 9. feb. 2024. [Na spletu]. Dostopno na: https://www.redhat.com/en/topics/devops/what-cicd-pipeline

[3] L. Bass, I. Weber, in L. Zhu, DevOps: A Software Architect's Perspective, 1. izd. Addison-Wesley Professional, 2015.

[4] O. Elazhary, C. Werner, Z. S. Li, D. Lowlind, N. Ernst, in M.-A. Storey, "Uncovering the Benefits and Challenges of Continuous Integration Practices," IEEE Trans. Softw. Eng., let. 48, št. 7, str. 2570-2583, jul. 2022, doi: 10.1109/TSE.2021.3064953.

[5] F. Zampetti, C. Vassallo, S. Panichella, G. Canfora, H. Gall, in M. Di Penta, "An empirical characterization of bad practices in continuous integration," Empir. Softw. Eng., let. 25, št. 2, str. 1095-1135, mar. 2020, doi: 10.1007/s10664-019-09785-8.

[6] J. Santos, D. A. da Costa, S. McIntosh, in U. Kulesza, "On the need to monitor continuous integration practices," Empir. Softw. Eng., let. 30, št. 5, str. 125, jun. 2025, doi: 10.1007/s10664-025-10682-6.

[7] M. Zarour, N. Alhammad, M. Alenezi in K. Alsarayrah, "A Research on DevOps Maturity Models," Int. J. Recent Technol. Eng. IJRTE, let. 8, št. 3, str. 4854-4862, sep. 2019, doi: 10.35940/ijrte.C6888.098319.

[8] J. A. V. M. K. Jayakody in W. M. J. I. Wijayanayake, "DevOps Maturity; A Systematic Literature Review," v 2024 International Research Conference on Smart Computing and Systems Engineering (SCSE), apr. 2024, str. 1-6. doi: 10.1109/SCSE61872.2024.10550493.

[9] P. Rostami Mazrae, T. Mens, M. Golzadeh, in A. Decan, "On the usage, co-usage and migration of CI/CD tools: A qualitative analysis," Empir. Softw. Eng., let. 28, št. 2, str. 52, mar. 2023, doi: 10.1007/s10664-022-10285-5.

[10] F. Zampetti, S. Geremia, G. Bavota, in M. Di Penta, "CI/CD Pipelines Evolution and Restructuring: A Qualitative and Quantitative Study," v 2021 IEEE International Conference on Software Maintenance and Evolution (ICSME), sep. 2021, str. 471-482. doi: 10.1109/ICSME52107.2021.00048.

[11] N. Chaudhari, "Pipelines Have Feelings Too: A Structured Way To Design CI/CD Pipelines," v Proceedings of the ACM/IEEE 27th International Conference on Model Driven Engineering Languages and Systems, MODELS Companion '24. New York, NY, USA: ACM, okt. 2024, str. 184-187. doi: 10.1145/3652620.3676876.

[12] J. Humble in D. Farley, Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation, 1. izd. Addison-Wesley Professional, 2010.

[13] D. Teixeira, R. Pereira, T. Henriques, M. Mira da Silva, J. Faustino, in M. Silva, "A maturity model for DevOps," Int. J. Agile Syst. Manag., let. 13, št. 4, str. 464-511, 2020, doi: 10.1504/IJASM.2020.112343.

[14] U. K. Anugula Sethupathy, "Navigating Continuous Improvement: An In-Depth Analysis of Lean-Agile and DevOps Maturity Models," J. Softw. Eng. Appl., let. 18, št. 9, str. 317-335, avg. 2025, doi: 10.4236/jsea.2025.189019.

[15] A. Kumar, M. Nadeem, in M. Shameem, "Prioritization of DevOps Maturity models using Fuzzy TOPSIS," v Proceedings of the 27th International Conference on Evaluation and Assessment in Software Engineering, EASE '23. New York, NY, USA: ACM, jun. 2023, str. 438-443. doi: 10.1145/3593434.3594241.

[16] N. Forsgren, J. Humble, in G. Kim, Accelerate: The Science of Lean Software and DevOps: Building and Scaling High Performing Technology Organizations. IT Revolution, 2018.

[17] D. DeBellis idr., "DORA | Accelerate State of DevOps Report 2024." Dostopno: 15. jul. 2026. [Na spletu]. Dostopno na: https://dora.dev

[18] J. Herbsleb, D. Zubrow, D. Goldenson, W. Hayes, in M. Paulk, "Software quality and the Capability Maturity Model," Commun. ACM, let. 40, št. 6, str. 30-40, jun. 1997, doi: 10.1145/255656.255692.

[19] "Whitepaper: A Roadmap to Continuous Delivery Pipeline Maturity." Dostopno: 7. avg. 2025. [Na spletu]. Dostopno na: https://pages.awscloud.com/awsmp-whitepaper-dev-roadmap-to-continuous-delivery-pipeline-maturity.html

[20] SLSA, "SLSA specification," SLSA. Dostopno: 15. jul. 2026. [Na spletu]. Dostopno na: https://slsa.dev/spec/v1.2/

[21] OpenSSF, "OpenSSF Scorecard - Open Source Security Foundation." Dostopno: 15. jul. 2026. [Na spletu]. Dostopno na: https://openssf.org/projects/scorecard/

[22] L. Marrero in H. Astudillo, "DevOps-RAF: An assessment framework to measure DevOps readiness in software organizations," v 2021 40th International Conference of the Chilean Computer Science Society (SCCC), nov. 2021, str. 1-8. doi: 10.1109/SCCC54552.2021.9650363.

[23] M. Faisal Abrar, A. Alferadi, T. S. Almurayziq, M. Saqib, W. Khan, in M. Alsafar, "Enhancing Extreme Programming (XP) Adoption Through SAMAM: A Scalable Agile Maturity Assessment Model Based on Industry Best Practices." [Na spletu]. Dostopno na: https://xplorestaging.ieee.org/document/11018324

[24] P. Hohl, M. Stupperich, J. Münch, in K. Schneider, "An Assessment Model to Foster the Adoption of Agile Software Product Lines in the Automotive Domain." [Na spletu]. Dostopno na: https://xplorestaging.ieee.org/document/8436325

[25] M. A. Akbar, S. Rafi, S. Hyrynsalmi, in A. A. Khan, "Towards People Maturity for Secure Development and Operations: A vision," v ACM Int. Conf. Proc. Ser., ACM, 2024, str. 528-533. doi: 10.1145/3661167.3661238.

[26] Y. Wang, M. V. Mantyla, Z. Liu, in J. Markkula, "Test automation maturity improves product quality - Quantitative study of open source projects using continuous integration," J. Syst. Softw., let. 188, str. 111259, jun. 2022, doi: 10.1016/j.jss.2022.111259.

[27] M. Zarour, N. Alhammad, M. Alenezi, in K. Alsarayrah, "Devops Process Model Adoption in Saudi Arabia: An Empirical Study," Jordanian J. Comput. Inf. Technol., let. 6, št. 3, str. 234-246, sep. 2020, doi: 10.5455/jjcit.71-1580581874.