02

UPORABNA INFORMATIKA

2025 ‹ ŠTEVILKA 2 ‹ LETNIK ????? ‹ ISSN 1318-1882

U P O R A B N A
I N F O R M A T I K A
2025 ŠTEVILKA 2 APR/MAJ/JUN LETNIK XXXIII ISSN 1318-1882

Znanstveni prispevki

Marko Kompara, Marko Hölbl
Vzpostavitev šifrirnih ključev v post-kvantni kriptografiji 55

Tilen Tratnjek, Gregor Polančič
Poslovanje pod drobnogledom - primerjalna analiza sodobnih orodij za rudarjenje procesov 80

Strokovni prispevki

Ester Bradač
Standardi in skladnost v IT projektih: integracija standardov ISO 27001/22301/9001 v vodenje projektov 95

Pregledni znanstveni prispevki

Mitja Gradišnik, Tina Beranič, Muhamed Turkanović
Analiza posebnosti zagotavljanja kakovosti pri razvoju decentraliziranih aplikacij v okolju Ethereum 102

Informacije

Iz Islovarja 115

2025 ŠTEVILKA 2 APR/MAJ/JUN LETNIK XXXIII ISSN 1318-1882

Ustanovitelj in izdajatelj
Slovensko društvo INFORMATIKA
Litostrojska cesta 54, 1000 Ljubljana

Predstavnik
Slavko Žitnik

Odgovorni urednik
Mirjana Kljajić Borštnar

Prelom in tisk
Boex DTP, d. o. o., Ljubljana

Naklada
110 izvodov

Naslov uredništva
Slovensko društvo INFORMATIKA
Uredništvo revije Uporabna informatika
Litostrojska cesta 54, 1000 Ljubljana
www.uporabna-informatika.si

Revija izhaja četrtletno. Cena posamezne številke je 20,00 EUR. Letna naročnina za podjetja 85,00 EUR, za vsak nadaljnji izvod 60,00 EUR, za posameznike 35,00 EUR, za študente in seniorje 15,00 EUR. V ceno je vključen DDV.

Revija Uporabna informatika je od številke 4/VII vključena v mednarodno bazo INSPEC.

Revija Uporabna informatika je pod zaporedno številko 666 vpisana v razvid medijev, ki ga vodi Ministrstvo za kulturo RS.

Revija Uporabna informatika je vključena v Digitalno knjižnico Slovenije (dLib.si).

Izid publikacije je finančno podprla Javna agencija za znanstvenoraziskovalno in inovacijsko dejavnost Republike Slovenije.

Vabilo avtorjem

V reviji Uporabna informatika objavljamo kakovostne izvirne prispevke domačih in tujih avtorjev z najširšega področja informatike, ki se nanašajo tako na poslovanje podjetij, javno upravo, družbo in posameznika. Prispevki so lahko znanstvene, strokovne ali informativne narave, še posebno spodbujamo objavo interdisciplinarnih prispevkov. Zato vabimo avtorje, da prispevke, ki ustrezajo omenjenim usmeritvam, pošljejo uredništvu revije po elektronski pošti na naslov ui@drustvo-informatika.si.

Avtorje prosimo, da pri pripravi prispevka upoštevajo navodila, ki so objavljena na naslovu http://www.uporabna-informatika.si.

Za kakovost prispevkov skrbi mednarodni uredniški odbor. Prispevki so anonimno recenzirani, o objavi pa na podlagi recenzij samostojno odloča uredniški odbor. Recenzenti lahko zahtevajo, da avtorji besedilo spremenijo v skladu s priporočili in da popravljeni prispevek ponovno prejmejo v pregled. Sprejeti prispevki so pred izidom revije objavljeni na spletni strani revije (predobjava), še prej pa končno verzijo prispevka avtorji dobijo v pregled in potrditev. Uredništvo lahko še pred recenzijo zavrne objavo prispevka, če njegova vsebina ne ustreza vsebinski usmeritvi revije ali če prispevek ne ustreza kriterijem za objavo v reviji.

Pred objavo prispevka mora avtor podpisati izjavo o avtorstvu, s katero potrjuje originalnost prispevka in dovoljuje prenos materialnih avtorskih pravic. Avtorji prejmejo enoletno naročnino na revijo Uporabna informatika, ki vključuje avtorski izvod revije in še nadaljnje tri zaporedne številke. S svojim prispevkom v reviji Uporabna informatika boste pomagali k širjenju znanja na področju informatike. Želimo si čim več prispevkov z raznoliko in zanimivo tematiko in se jih že vnaprej veselimo

Uredništvo revije

Navodila avtorjem člankov

Članke objavljamo praviloma v slovenščini, članke tujih avtorjev pa v angleščini. Besedilo naj bo jezikovno skrbno pripravljeno. Priporočamo zmernost pri uporabi tujk in, kjer je mogoče, njihovo zamenjavo s slovenskimi izrazi. V pomoč pri iskanju slovenskih ustreznic priporočamo uporabo spletnega terminološkega slovarja Slovenskega društva Informatika, Islovar (www.islovar.org).

Znanstveni prispevek naj obsega največ 40.000 znakov, kratki znanstveni prispevek do 10.000 znakov, strokovni članki do 30.000 znakov, obvestila in poročila pa do 8.000 znakov.

Prispevek naj bo predložen v urejevalniku besedil Word (*.doc ali *.docx) v enojnem razmaku, brez posebnih znakov ali poudarjenih črk. Za ločilom na koncu stavka napravite samo en presledek, pri odstavkih ne uporabljajte zamika.

Naslovu prispevka naj sledi polno ime vsakega avtorja, ustanova, v kateri je zaposlen, naslov in elektronski naslov. Sledi naj povzetek v slovenščini v obsegu 8 do 10 vrstic in seznam od 5 do 8 ključnih besed, ki najbolje opredeljujejo vsebinski okvir prispevka. Sledi naj prevod naslova povzetka in ključnih besed v angleškem jeziku. V primeru, da oddajate prispevek v angleškem jeziku, velja obratno. Razdelki naj bodo naslovljeni in oštevilčeni z arabskimi številkami.

Slike in tabele vključite v besedilo. Opremite jih z naslovom in oštevilčite z arabskimi številkami. Na vsako sliko in tabelo se morate v besedilu prispevka sklicevati in jo pojasniti. Če v prispevku uporabljate slike ali tabele drugih avtorjev, navedite vir pod sliko oz. tabelo. Revijo tiskamo v črno-beli tehniki, zato barvne slike ali fotografije kot original niso primerne. Slikam zaslonov se v prispevku izogibajte, razen če so nujno potrebne za razumevanje besedila. Slike, grafikoni, organizacijske sheme ipd. naj imajo belo podlago. Enačbe oštevilčite v oklepajih desno od enačbe.

V besedilu se sklicujte na navedeno literaturo skladno s pravili sistema IEEE navajanja bibliografskih referenc, v besedilu to pomeni zaporedna številka navajenega vira v oglatem oklepaju (npr. [1]). Na koncu prispevka navedite samo v prispevku uporabljeno literaturo in vire v enotnem seznamu, urejeno po zaporedni številki vira, prav tako v skladu s pravili IEEE. Več o sistemu IEEE, katerega uporabo omogoča tudi urejevalnik besedil Word 2007, najdete na strani https://owl.purdue.edu/owl/research_and_citation/ieee_style/ieee_general_format.html.

Prispevku dodajte kratek življenjepis vsakega avtorja v obsegu do 8 vrstic, v katerem poudarite predvsem strokovne dosežke.

Vzpostavitev šifrirnih ključev v post-kvantni kriptografiji

Marko Kompara, Marko Hölbl
Fakulteta za elektrotehniko, računalništvo in informatiko, Univerza v Mariboru
marko.kompara@um.si, marko.holbl@um.si

Izvleček

Razvoj kvantnega računalnika napreduje tako hitro, da čeprav kriptografsko signifikanten kvantni računalnik še ne obstaja in ga po najbolj optimističnih napovedih lahko pričakujemo komaj čez kakšno desetletje, je že potrebna zamenjava določenih ranljivih kriptografskih gradnikov, ki se danes uporabljajo za varovanje podatkov. Ta prispevek je namenjen pregledu mehanizmov za inkapsulacijo ključa (KEM), ki so kvantno varni algoritmi za dogovor o šifrirnih ključih. V prispevku se bomo dotaknili nevarnosti kvantnih računalnikov za tradicionalno kriptografijo, predstavili splošno delovanje algoritmov KEM ter specifične rešitve, za katere bomo predstavili njihove varnostne lastnosti ter izvedli analizo učinkovitosti oz. hitrosti njihovega delovanja. Na koncu bomo predstavili še nekaj primerov uporabe algoritmov KEM oz. hibridnih rešitev, ki združujejo tradicionalno in post-kvantno kriptografijo v pogosto uporabljenih protokolih.

Ključne besede: KEM, mehanizem za inkapsulacijo ključa, post-kvantna kriptografija, analiza.

Establishing Encryption Keys with Post-Quantum Cryptography

Abstract

The development of the quantum computer is advancing so fast that, although a cryptographically significant quantum computer does not yet exist and, according to the most optimistic forecasts, can only be expected in about a decade, some of the vulnerable cryptographic building blocks used to protect data today need to be replaced now. This paper is an overview of key encapsulation mechanisms (KEMs), which are quantum-secure algorithms for key agreement. In the paper, we will discuss the threat quantum computers pose to traditional cryptography, present the general operation of KEM algorithms and specific algorithms for which we will present their security properties, and perform an analysis of their performance. Finally, we will present some examples of KEM algorithms or hybrid solutions that combine traditional and post-quantum cryptography in commonly used protocols.

Keywords: KEM, key encapsulation mechanism, post-quantum cryptography, analysis.

1 UVOD

Čeprav ideja kvantnega računalništva obstaja že več desetletij, je bila njihova realna implementacija dolgo vprašljiva. Vendar je, z velikimi vložki denarja in raziskovalnega truda, ta v zadnjih letih postala dosegljiva resničnost. Kvantni računalniki so problem za kibernetsko varnost od leta 1994 [1], ko je Peter Shor odkril kvantni algoritem (od takrat imenovan Shorov algoritem), ki bi lahko, ko bodo kvantni računalniki postali dovolj močni, razbil tradicionalno asimetrično kriptografijo oz. kriptografijo z javnim ključem (npr. RSA, ECC), ki je temelj sodobne kibernetske varnosti. To je botrovalo nastanku post-kvantne kriptografije (angl. Post-Quantum Cryptography), ki zahteva, da so kriptografski elementi (poudarek je predvsem na algoritmih javnega ključa) varni pred kriptoanalitičnimi napadi tradicionalnega računalnika in kvantnega računalnika. Medtem ko kvantni

računalniki, ki so dovolj zmogljivi, da bi izničili varnost tradicionalnih asimetričnih algoritmov, še niso na voljo, je strategija napada SNDL (angl. Store-Now Decrypt-Later) že zaskrbljujoča in jo je treba nasloviti. Z napadom NSDL napadalec shrani vse šifrirane podatke, ki jih lahko zajame sedaj, v upanju, da jih bo dešifriral, ko bodo kvantni računalniki postali dovolj močni in dostopni.

Nacionalni inštitut za standarde in tehnologijo iz ZDA - NIST (angl. National Institute of Standards and Technology) se že skoraj desetletje pripravlja na posledice prihajajočih kvantnih računalnikov. NIST se tudi pripravlja na ukinitev uporabe tradicionalnih asimetričnih algoritmov do leta 2035 [2] in večina Evropskih držav, vključno s Slovenijo je podala skupno izjavo o namenu prehoda na post-kvantno kriptografijo do konca leta 2030 [3]. NIST je zato že leta 2016 začel tekmovanje za standardizacijo post-kvantne kriptografije, da bi poiskal nove rešitve, ki bi nadomestile tradicionalno asimetrično kriptografijo, ki jo trenutno uporabljamo in ki bo razbita pod kriptografsko relevantnim kvantnim računalnikom (angl. Cryptographically-relevant quantum computer). NIST je že izbral in tudi standardiziral nekatere post-kvantne algoritme za elektronsko podpisovanje in izmenjavo ključev [4]. V tem članku se bomo osredotočili na algoritme za izmenjavo ključa.

Šifriranje javnega ključa – PKE (angl. Public Key Encryption), lahko šifrira poljubno sporočilo, medtem ko mehanizem za inkapsulacijo ključa - KEM (angl. Key Encapsulation Mechanism) inkapsulira naključno kratko skrivnost (javni in zasebni ključi se uporabljajo za inkapsulacijo in dekapsulacijo). PKE se lahko uporablja za izmenjavo ključev (ključ/skrivnost se ustvari ločeno in nato šifrira). Primer takšne uporabe je algoritem RSA. Algoritmi KEM so kot omejeni PKE algoritmi, ki pa imajo prednost lažjega dokazovanja varnosti, so učinkovitejši in ne potrebujejo bitnega zapolnjevanja (angl. Padding). Algoritme KEM je mogoče izdelati iz PKE, kar tudi drži za vse algoritme KEM omenjene v tem prispevku. Prvi del tega poročila bo posvečen pregledu pomembnih post-kvantnih algoritmov KEM, njihovi varnosti in učinkovitosti.

Kako zmogljiv mora biti kriptografsko relevanten kvantni računalnik (tj. dovolj zmogljiv, da lahko razbije tradicionalno asimetrično kriptografijo), se s časom spreminja. Leta 2015 so raziskovalci ocenili, da bi kvantni računalnik potreboval milijardo (10^9) kubitov, da bi razbil 2048-bitni RSA [5]. To se je leta 2021 zmanjšalo na samo 20 milijonov kubitov, kar bi trajalo približno osem ur. Trenutno sta dva kvantna računalnika z največ qubiti razvila IBM in Atom Computing s 1121 in 1180 kubiti. Vendar pa niso vsi kubiti enaki, koliko šuma in kako učinkovito je ta šum mogoče izničiti, močno vpliva na moč kvantnega računalnika. IBM načrtuje, da bo do leta 2033 imel superračunalnik s sto tisoč kubiti [6].

Post-kvantni algoritmi KEM, ki so najbolj znani in temeljito preizkušeni, so šli skozi NIST-ovo tekmovanje. Natečaj je vključeval tudi algoritme za elektronske/digitalne podpise, vendar se bomo v tem prispevku osredotočili na rešitve KEM. Čeprav so na voljo novi in standardizirani algoritmi elektronskega podpisa, je prehod infrastrukture javnih ključev – PKI (angl. Public Key Infrastructure) na uporabo post-kvantne kriptografije težji in bo trajal več časa, kot prehod algoritmov za dogovor o ključu. Preverjanje pristnosti je izjemno pomembno za preprečevanje napadov s posrednikom - MitM (angl. Man-in-the-Middle). Na srečo, za razliko od tradicionalnih rešitev za dogovor o ključu, ki trpijo zaradi strategije napada SNDL in jih je treba obravnavati čim prej, preverjanje pristnosti nima enakih težav. Ena od glavnih aplikacij, za katere se uporabljajo elektronski podpisi, je zagotavljanje kratkotrajnega overjanja (podpis se ustvari, kmalu zatem overi in nato se nikoli več ne uporabi). Primer tega je vzpostavljanje povezave TLS. Takšnih primerov uporabe ni potrebno takoj spremeniti v kvantno odporno obliko, ker sposobnost ponovnega ustvarjanja podpisov v prihodnosti s kvantnimi računalniki ne izničuje varnosti preverjanja pristnosti danes in današnji podpis v prihodnosti ne bo imel nobene vrednosti (ker je uporaben samo kratkoročno). To pomeni, da post-kvantno overjanje še ni tako tako nujno in jih bo treba algoritme spremeniti šele, ko bo na voljo kvantni računalnik (ali ko bo verjetnost, da takšen računalnik obstaja, dovolj visoka). Medtem so trenutne rešitve, ki temeljijo na RSA in ECC, varne in bolj priročne. Seveda, vse to, velja samo za kratkoročno preverjanje pristnosti, situacija je nekoliko bolj zapletena za primere dolgoročne uporabe (npr. uporaba elektronskih podpisov v pogodbah ali verigah blokov (angl. Blockchain)). Medtem ko so algoritmi post-kvantnega podpisa standardizirani s strani NIST (npr. ML-DSA, SLH-DSA in prihajajoči FN-DSA) in obstaja nekaj protokolov za njihovo izvajanje za namene

overjanja entitet v komunikaciji (npr. [7], [8], [9]), se trenutno ne sprejemajo množično, ker so kvantni certifikati veliko večji od tipičnih potrdil, kar povzroča slabo učinkovitost in težave z združljivostjo z nekaterimi omrežnimi napravami.

Zadnji del tega prispevka bo namenjen rešitvam, ki uporabljajo obravnavane post-kvantne algoritme KEM. Imamo že dolgo uveljavljene protokole (npr. TLS), ki uporabljajo tradicionalno asimetrično kriptografijo za izmenjavo ključev in nato uporabljajo simetrične algoritme za šifriranje vseh izmenjanih podatkov. Kot je že bilo omenjeno, kvantni računalniki ne izničijo varnosti simetrične kriptografije. Za prehod obstoječih protokolov na kvantno varne algoritme bi zato bilo treba spremeniti/zamenjati tradicionalne asimetrične algoritme, ki se uporablja za izmenjavo ključev (in algoritme za overjanje, ki pa niso del tega prispevka). Eden od načinov za to bi bil, da se ti algoritmi preprosto zamenjajo s post-kvantnimi KEM. Ker so algoritmi KEM odporni na napade tradicionalnih računalnikov in kvantnih računalnikov, bodo nastali protokoli varni. Vendar pa je varnostna skupnost zelo nenaklonjena temu, ker so algoritmi KEM relativno novi, niso bili preizkušeni v realnih okoljih in niso dolgoročno dokazali svoje varnosti. Strah je, da bi revolucionarni napad na njihovo izvajanje ali osnovni problem, na katerem temeljijo povzročil, da bi specifični algoritmi KEM čez noč postali nevarni. Ta strah ni več prisoten pri trenutno uporabljenih algoritmih, ker se uporabljajo že zelo dolgo in v tem času niso bile ugotovljene nobene pomembne pomanjkljivosti. Iskanje pravega trenutka za prehod iz tradicionalnih na nove metode vzpostavitve ključev je zato problematično. Previden pristop je združevanje tradicionalnih in algoritmov KEM na način, da je treba razbiti oba algoritma, da bi razkrili zaščiteno skrivnost (tj. izmenjano skupno skrivnost/ključ). Na ta način zaščiteni podatki ne bodo razkriti, ko kvantni računalniki postanejo dovolj močni, da razbijejo tradicionalne algoritme, ker bo post-kvantni KEM odporen, in če pride do težav z novejšim algoritmom KEM, bo robusten tradicionalni algoritem preprečil najhujše posledice (tj. ne bodo takoj zlomljivi, vsaj ne, dokler ne bo obstajal kriptografsko relevantni kvantni računalnik). Te vrste mehanizmov, ki uporabljajo obe vrsti KEM, imenujemo post-kvantni/tradicionalni hibridi, pogosto samo hibridi. Poleg večjega zaupanja v moč kombiniranih algoritmov (ker bi moral napadalec razbiti oba) hibridni pristop podpira tudi enostavno in postopno (različne storitve bodo imele možnost migracije ob različnih časih brez težav) migracijo na popolnoma post-kvantne rešitve, ko bo post-kvantna kriptografija dovolj zrela. Edina slabost hibridne sestave, je bolj zahtevno oz. počasnejše izvajanje. V zadnjem delu tega prispevka bo predstavljena konstrukcija in primeri takšnih hibridnih algoritmov KEM in protokolov, ki jih uporabljajo.

2 POST-KVANTNI MEHANIZMI ZA INKAPSULACIJO KLJUČA (KEM)

KEM kandidati na NIST tekmovanju so bili predloženi novembra 2017 in od takrat so kandidati opravili strogo varnostno testiranje in testiranje učinkovitosti v več ciklih v katerih so se algoritmi tudi nekoliko spreminjali. Trenutno edini kandidat, ki je bil standardiziran je ML-KEM. ML-KEM je standardizirana različica (z nekaj zelo majhnimi spremembami) algoritma CRYSTALS-Kyber (v nadaljevanju samo Kyber). Kyber je bil izbran za standardizacijo s strani NIST, ker je bila to (po njihovem mnenju) najboljša rešitev, predvsem iz vidika učinkovitosti. NIST zatem odprl četrti krog tekmovanja za izbiro in standardizacijo enega ali več dodatnih algoritmov KEM [10].

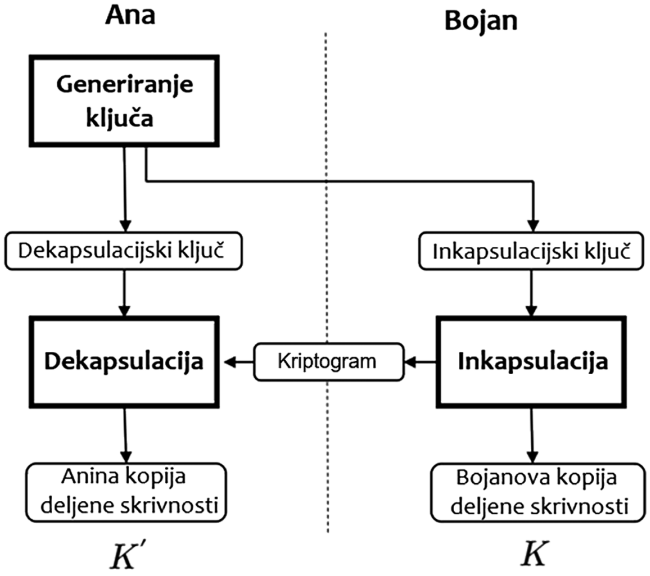
Prvi in za zdaj edini od algoritmov, ki so se uvrstili v četrti krog, za katerega je NIST napovedal standardizacijo je HQC [11]. Algoritem sicer še ni bil standardiziran, zato se bomo v tem prispevku še sklicevali na nestandardiziran HQC. Dva dodatna algoritma, ki jih bomo vključili v ta pregled sta, še vedno kandidata za standardizacijo (NIST ni napovedal, ali ima namen standardizirati še kakšno rešitev). To sta Classic McElice in BIKE. Edini algoritem, ki je vključen v četrti krog NIST tekmovanja, ki ga ne vključujemo v pregled je algoritem SIKE, ker je bil problem na podlagi katerega zagotavlja varno delovanje (tj. Izogeneza supersingularne eliptične krivulje) od začetka četrtega kroga dokazano razbit. Zadnji algoritem, o katerem bomo razpravljali tudi v tem prispevku je FrodoKEM, ki se kljub temu, da je bil del tekmovanja NIST, ni prebil v četrti krog. Čeprav NIST ni izbral algoritma FrodoKEM, ga nemški BSI [12] in francoski ANSSI [13] podpirata. Tabela 1 navaja algoritme KEM, obravnavane v tem dokumentu, skupaj z njihovim statusom (kandidati četrtega kroga tekmovanja NIST so še vedno lahko standardizirani s strani NIST) in iz katerega tipa post-kvantne kriptografije izhajajo.

Tabela 1: **Algoritmi KEM, obravnavani v tem prispevku.**

Algoritem KEM	Stanje	Vrsta
ML-KEM [14]	Standardiziran	Mreža (angl. Lattice)
HQC [15]	Izbran za standardizacijo	Koda (angl. Code)
Classic McEliece [16]	Kandidat v četrtem krogu	Koda (angl. Code)
FrodoKEM [17]	Nadomestni kandidat tretjega kroga	Mreža (angl. Lattice)
BIKE [18]	Kandidat v četrtem krogu	Koda (angl. Code)

Vredno je omeniti še algoritma Saber in NTRU, ki sta se prav tako uvrstila v tretji krog tekmovanja in sta se zelo dobro odrezala. NIST se je odločil, da jih ne bo standardiziral (enako kot FrodoKEM), ker temeljijo na isti osnovni strukturi kot Kyber/ML-KEM (tj. strukturirana mreža). NIST meni, da je Kyber najboljša rešitev od treh in ne želijo standardizirati več algoritmov, ki temeljijo na istem problemu, ker bi ranljivost v tem osnovnem problemu lahko povzročila razbitje vseh takšnih algoritmov. Iz istega razloga želi NIST standardizirati vsaj še en algoritem (to bo HQC), ki sloni na drugačnem osnovnem problemu. To zagotavlja, da če se v prihodnosti najde rešitev za problem mrež (zaradi česar bo ML-KEM neuporaben), bo rešitev preprost prehod na alternativen standardiziran KEM (Saber in NTRU ne bi bila dobra izbira, ker bi bila najverjetneje razbita istočasno). Hkrati se, zaradi preprostosti in lažje združljivosti, nabor standardiziranih algoritmov ohranja majhen (tj. se ne standardizira več algoritmov kot je potrebno). Glede na to, da Saber in NTRU ne bosta standardizirana in trenutno ni znakov, da imata podporo drugje, nista bila vključena v ta prispevek. FrodoKEM ni bil izbran s strani NIST iz istega razloga, vendar je FrodoKEM z varnostnega vidika bolj konzervativna rešitev (ne uporablja strukturiranih mrež, ampak mreže, ki niso strukturirane), zato ga nekateri smatrajo, kot dobro alternativo za ML-KEM.

Vsi algoritmi KEM so sestavljeni iz treh splošnih funkcij: generiranje ključev, inkapsulacija in dekapsulacija. Generiranje ključev nima vhodnih parametrov in ustvari zasebni (imenovan dekapsulacijski ključ) in javni ključ (imenovan inkapsulacijski ključ). Javni ključ je poslan entiteti s katero želimo izmenjati skrivnost. Inkapsulacijo izvede prejemnik javnega ključa. Inkapsulacija uporabi javni ključ kot vhod, ustvari naključni skrivni ključ (ali skupno skrivnost, iz katere je mogoče ustvariti ključ) in vrne ključ/skrivnost in njeno inkapsulirano vrednost - tj. kriptogram (angl. Ciphertext). Kriptogram se pošlje nazaj imetniku zasebnega ključa (tj. entiteti, ki je prvotno ustvarila par ključev). Dekapsulacija vzame kot vhod vrednosti zasebnega ključa in kriptograma. Vrne vrednost deljene skrivnosti/ključa ali pa ne uspe (to se lahko zgodi, če so uporabljeni napačni ključi in včasih, čeprav zelo redko, se lahko zgodi tudi pri uporabi ustreznih ključev - v tem primeru je treba postopek ponoviti). Postopek delovanja algoritmov KEM je predstavljen spodaj (Slika 1).



Slika 1: **Prikaz delovanja algoritmov KEM po FIPS 2023, povzeto po [14].**

2.1 ML-KEM

ML-KEM (Module-Lattice-Based Key-Encapsulation Mechanism) je trenutno edini standardiziran post-kvantni KEM. Standardiziran je bil v FIPS 203 (Federal Information Processing Standards) [14]. ML-KEM temelji na CRYSTALIS-Kyber iz tretjega kroga tekmovanja NIST, vendar vključuje nekaj manjših sprememb (npr. fiksna velikost deljenega skrivnega ključa in manj uporabljenih zgoščevalnih operacij).

ML-KEM ima hitro generiranje ključev, inkapsulacijo in dekapsulacijo (v primerjavi z drugimi primerljivimi algoritmi), velikost javnega ključa in kriptogram pa predstavljata dobro ravnovesje med varnostjo in učinkovitostjo [19].

Varnost ML-KEM temelji na problemu modularnega učenja z napakami (angl. Module Learning with Errors - MLWE) v modularnih mrežah (angl. Modular lattice) [20]. Trenutno ni znakov, da je ta metoda vzpostavitve skupne skrivnosti nevarna v tradicionalnih ali kvantnih računalnikih. ML-KEM je zgrajen v dveh korakih. Problem MLWE se uporablja za ustvarjanje algoritma šifriranja javnega ključa (angl. Public-Key Encryption - PKE). PKE se nato pretvori z uporabo Fujisaki-Okamoto transformacije [21] v mehanizem za inkapsulacijo ključev (KEM). Nastali KEM zagotavlja varnost v skladu s splošnejšim modelom napada kot osnovni algoritem PKE, kar ima za posledico varnost IND-CCA2 [22].

ML-KEM je definiran s tremi nabori parametrov: ML-KEM-512, ML-KEM-768 in ML-KEM-1024. NIST priporoča uporabo ML-KEM-768 kot privzetega nabora parametrov, saj zagotavlja veliko varnostno maržo ob sprejemljivi hitrosti izvajanja [14]. ML-KEM je med vsemi kandidati za standardizacijo bil prepoznaven po svoji hitrosti delovanja in velikosti kriptogramov ter ključev.

Neuspeh dekapsulacije (angl. Decapsulation failure) je, ko postopek dekapsulacije ne vrne istega skrivnega ključa, kot je bil ustvarjen v inkapsulaciji, čeprav so bili vhodi pravilni in je bilo generiranje naključnosti uspešno. Verjetnost, da se to zgodi, pa je izjemno majhna. Verjetnost neuspeha dekapsulacije (angl. decapsulation failure rate - DFR) ML-KEM je [14]:

- $2^{-138,8}$ za ML-KEM-512,
- $2^{-164,8}$ za ML-KEM-768 in
- $2^{-174,8}$ za ML-KEM-1024.

ML-KEM in vsi algoritmi KEM opisani v tem prispevku uporabljajo naključne vrednosti, ustvarjene pri njihovem generiranju ključev (ali fazi pred njim). Ker funkcija generiranja ključev ne sprejema nobenih parametrov, so te naključne vrednosti tiste, ki v celoti določajo, kakšen par ključev se ustvari. Standard ML-KEM omogoča, da se dve (32 zlogov veliki) naključni vrednosti shranita kot seme namesto zasebnega ključa. Ko je zasebni ključ potreben, se lahko uporabi notranja funkcija, ki generira ključ iz semena. Ta možnost močno zmanjša količino podatkov, ki jih je treba shraniti kot zasebni/dekapsulacijski ključ (64 namesto 1632, 2400 ali 3168 zlogov, odvisno od nabora parametrov) in zagotavlja dodatno varnostno lastnost (kot bo omejeno v poglavju 3.1.2). Slaba stran shranjevanja semena namesto zasebnega ključa je dodatno procesiranje; vendar je razlika v obdelavi za razširitev zasebnega ključa ML-KEM minimalna, medtem ko je uporaba semen veliko enostavnejša. Obstajajo že pobude, da bi uporaba semen postala de-facto način implementacije ML-KEM [23]. Nekaj podobnega bi se lahko implementiralo tudi z drugimi algoritmi KEM, vendar uradno tega trenutno ne podpirajo. Drugi algoritmi imajo večje zasebne ključe in počasneje ustvarjajo pare ključev, zato bi bil kompromis med uporabo pomnilnika in hitrostjo večji.

2.2 Classic McEliece

Classic McEliece [16], [24] je konzervativna rešitev na problem, ki ga predstavlja kvantni računalnik, ker temelji na kriptosistemu McEliece, ki je bil prvič predlagan leta 1978 [25]. To pomeni, da je sistem in problem na katerem sloni algoritem imel veliko časa za dozorevanje in strokovnjaki, so imeli veliko časa, da bi v njem našli ranljivosti. Iz tega razloga, je ta algoritem tudi veljal kot favorit za standardizacijo, kot alternativna izbira algoritmu ML-KEM.

Classic McEliece KEM je zgrajen iz PKE, ki je narejen iz Niederreiterjeve dvojne različice McEliecejevega PKE z uporabo binarnih kod Goppa in preko posebej modificirane Fujisaki-Okamoto transformacije pretvorjen v KEM, da doseže dodatno učinkovitost in varnost IND-CCA2 [26], [27]. Classic McEliece je tudi v procesu ISO standardizacije [28], [29].

Obstaja več različic algoritma Classic McEliece. Tu bomo omenili tri najpomembnejše. Prva je označena kot različica pc (Plaintext Confirmation). To je bila različica Classic McEliece, ki je bila predložena v NIST tekmovanje v krogih 1 do 3 [16], [30]. V 4. krogu je bil nadomeščen z različicama f in ne-f (tj. za naborom parametrov ni oznake). Različica pc je imela daljše kriptograme in je bila varnejša, a tudi počasnejša. Različice f imajo hitrejše generiranje ključev, medtem ko imajo različice brez oznake, enostavnejše generiranje ključev [31]. Različne različice algoritma uporabljajo različne parametre in niso interoperabilne. Za različici f in ne-f Classic McEliece obstaja pet izbranih naborov parametrov (ti so predstavljeni kasneje v članku).

Glavna prednost algoritma Classic McEliece je njegova učinkovita inkapsulacija in dekapsulacija. Po drugi strani pa je generacija ključev zelo počasna, kar bi lahko predstavljalo težavo pri komunikaciji v realnem času. Druga pomanjkljivost je velikost ključev, zlasti javnega (tj. inkapsulacijskega) ključa. Glede na nabor parametrov je lahko ta do 1,3 MB velik. To bi lahko bila velika težava v omejenih okoljih z omejenimi viri (npr. vgrajene naprave). Glede na slabosti in prednosti algoritma Classic McEliece in dejstva, da so njegovi pari ključev varni za ponovno uporabo, se zdi, da je najučinkovitejši način uporabe tega algoritma ponovna uporaba para ključev in izogibanje režijskim stroškom povezanih z generiranjem ključev in pošiljanja javnega ključa.

2.3 FrodoKEM

FrodoKEM [17] je družina algoritmov, katerih varnost izhaja iz previdne parametrizacije dobro raziskanega problema učenja z napakami (LWE), ki ima tesno povezavo z domnevno težkimi problemi na generičnih, algebrsko nestrukturiranih mrežah (angl. unstructured lattices) [17]. To je nekoliko povezano z ML-KEM, vendar ML-KEM uporablja strukturirane mreže (obstaja struktura/simetrija LWE). Strukturirane mreže ponujajo boljšo učinkovitost delovanja, manjše kriptograme in manjše ključe. Čeprav ni bila dokazana nobena razlika v stopnji varnosti, se nekateri strokovnjaki bojijo, da bi lahko bile strukturirane mreže šibka točka kriptografskega sistema in zato podpirajo bolj konzervativno rešitev, ki je FrodoKEM, kot alternativo [12]. Od rešitev predstavljenih v tem prispevku sta Classic McEliece in FrodoKEM dve najbolj konzervativni (tj. najbolj zreli in najmanj verjetno, da bi vsebovali ranljivosti) rešitvi.

Kot vsi drugi KEM v tem dokumentu je FrodoKEM zgrajen iz PKE (FrodoPKE) z dokazljivo varnostjo IND-CPA, ki se pretvori, podobno kot ML-KEM, z uporabo različice transformacije Fujisaki-Okamoto (natančneje HHK), v KEM z varnostjo IND-CCA2 [32]. Za svoje delovanje FrodoKEM uporablja psevdonaključni generator za generiranje javne matrike A, ki jo je mogoče izvesti z uporabo algoritmov AES-128 [33] ali SHAKE128 [34] (to razlikovanje bo postalo pomembno pri merjenju zmogljivosti).

Od tretjega kroga tekmovanja NIST je bil FrodoKEM posodobljen in je v postopku standardizacije v reviziji ISO/IEC 18033-2 [28], [35]. Kot del najnovejše posodobitve je bil FrodoKEM razdeljen na dve različici [17]. Med obema so majhne razlike (npr. FrodoKEM uporablja sol (angl. Salt) in ima večja semena), vendar je glavna razlika med njimi, da je pri FrodoKEM par ključev mogoče prosto ponovno uporabiti, medtem ko je pri eFrodoKEM treba par ključev spremeniti po vsaj vsakih 256 uporabah, kar preprečuje določene oblike napadov [17]. Ta razdelitev na dve različici lahko povzroči zmedo, ker je bila do vključno tretjega kroga tekmovanja NIST predložena samo ena različica algoritma. Ta različica je bila poimenovana FrodoKEM, vendar se je sedaj preoblikovala v eFrodoKEM. Pri pregledovanju starejših zapisov o algoritmu FrodoKEM je zato pomembno upoštevati, da se starejše informacije (pred razdelitvijo) imenujejo FrodoKEM, vendar se sklicujejo na trenutno specifikacijo eFrodoKEM, in to, kar je zdaj FrodoKEM, pred razdelitvijo ni obstajalo.

FrodoKEM ima tri nabore parametrov in dva možna algoritmom za generiranje matrike A. Skupaj ti predstavljajo šest naborov: FrodoKEM-640-AES, FrodoKEM-976-AES, FrodoKEM-1344-AES, FrodoKEM-640-SHAKE, FrodoKEM-976-SHAKE in FrodoKEM-1344-SHAKE. Enake kombinacije obstajajo za eFrodoKEM.

Neuspeh pri dekapsulaciji je, tako kot pri vseh tukaj predstavljenih post-kvantnih algoritmih KEM, možen tudi v FrodoKEM, čeprav je zelo malo verjetno [17], tudi v primerjavi z ML-KEM (enake vrednosti veljajo za eFrodoKEM):

- $2^{-138,7}$ za FrodoKEM-640 (AES ali SHAKE),
- $2^{-199,6}$ za FrodoKEM-976 (AES ali SHAKE) in
- $2^{-252,5}$ za FrodoKEM-1344 (AES ali SHAKE).

Glavna prednost algoritma FrodoKEM je njegova navidezna robustnost, ki izhaja iz uporabe dobro uveljavljenega in razumljenega problema LWE. Po drugi strani pa ima nadpovprečno velik javni ključ in največji kriptogram in zasebni ključ od vseh predstavljenih algoritmov KEM. Hitrost delovanja je posledično slabša.

2.4 BIKE

BIKE (Bit-flipping Key Encapsulation) [18] je algoritem KEM, ki temelji na vrsti kod za popravljanje napak QC-MDPC (Quasi Cyclic Moderate Density Parity Check). Tako kot Classic McEliece temelji na sistemu McEliece [25], vendar uporablja druge kode. Kode QC-MDPC so cenjene zaradi svoje učinkovitosti pri popravljanju napak in se v veliki meri upora-

bljajo v kriptografiji (med drugim jih uporablja tudi algoritem HQC).

Kot vsi ostali opisani algoritmi KEM, tudi BIKE uporablja različico transformacije Fujisaki-Okamoto, da naredi IND-CCA2 varen KEM iz PKE [18], [36]. Vendar pa lahko BIKE doseže IND-CCA2 le, če je mogoče doseči dovolj nizko verjetnost neuspeha dekapsulacije algoritma (DFR). Čeprav se zdi, da je DFR v praksi dovolj majhen, avtorji uradno ne trdijo, da je dosežena varnost IND-CCA2. Nekatere nedavne raziskave podpirajo idejo, da bi lahko bil BIKE INDCCA2 varen (z manjšimi spremembami specifičnih parametrov algoritma) [37]. BIKE torej uradno dosega samo varnost IND-CPA. BIKE se zato lahko uporablja le z efemernimi (tj. kratkotrajnimi) ključi, kar pomeni, da se za vsako sejo ustvari nov par ključev in dekapsulacija več kot enega kriptograma (z istim parom ključev) ni dovoljena. BIKE je bil zasnovan za uporabo s sinhronimi komunikacijskimi protokoli (npr. TLS) s kratkotrajnimi ključi, kjer je potreben (in zadosten) samo IND-CPA [18]. BIKE se lahko uporablja tudi z dolgotrajnimi ali statičnimi pari ključev (čeprav tega prvotno ni podpiral), vendar morajo biti v tem primeru izpolnjene predpostavke o dekoderju, da se zagotovi IND-CCA2.

BIKE je definiran s tremi nabori parametrov, ki ustrezajo trem različnim NIST varnostnim ravnem. Običajno so označeni kot BIKE-L1 (L je za raven (angl. Level)), BIKE-L3 in BIKE-L5.

Prednost BIKE je relativno majhna velikost komunikacijskih podatkov v primerjavi z drugimi algoritmi KEM. Pri uporabi s kratkotrajnimi ključi samo ML-KEM pošlje manj podatkov po ‚žici' (ko se ključi uporabljajo dolgoročno, postane Classic McEliece učinkovitejši zaradi majhnih kriptogramov). Ko pride do hitrosti delovanja pa je primerljiv s HQC, čeprav je tipično malo počasnejši.

2.5 HQC

Hamming Quasi-Cyclic (HQC) [15] je KEM, ki temelji na strukturiranih kodah (angl. structured codes). Zaradi tega je podoben Classic McEliece in še posebej algoritmu BIKE, ker se tudi zanaša na kode QC-MDPC (Quasi Cyclic Moderate Density Parity Check). Shema šifriranja javnega ključa HQC PKE se pretvori v HQC KEM s transformacijo HHK [38] (enako kot FrodoKEM) [15]. To, tako kot pri drugih algoritmih, daje HQC IND-CCA2 varnost, ker je za razliko od BIKE pokazan dovolj majhen DFR.

HQC ponuja tri nabore parametrov, HQC-128, HQC-192 in HQC-256, poimenovane po nivojih varnosti, ki jih zagotavljajo. Po zmogljivost je HQC primerljiv z BIKE, z nekoliko večjimi velikostmi parametrov, vendar nekoliko boljšo računalniško hitrostjo.

HQC, je drugi algoritem, ki je bil pred kratkim izbran za standardizacijo s strani NIST, vendar trenutno še nima standardizirane oblike. Čeprav bo tudi HQC standardiziran, NIST primarno priporoča uporabo algoritma ML-KEM in implementacijo HQC, samo kot rezervo v primeru, da se v ML-KEM odkrijejo ranljivosti.

3 PRIMERJAVA POST-KVANTNIH ALGORITMOV KEM

Za izbiro najboljšega algoritma KEM je treba upoštevati vidika varnosti in zmogljivosti algoritmov. Pri obravnavi zmogljivosti algoritmov mislimo predvsem na računsko kompleksnost (več o tem v nadaljevnaju), vendar je velikost javnega (tj. inkapsulacijskega) ključa, zasebnega (tj. dekapsulacijskega) ključa, kriptograma in deljene skrivnosti/ključa prav tako del učinkovitosti. Ti podatki določajo, koliko podatkov bo treba shraniti na napravah in koliko podatkov bo treba prenesti. To je lahko zelo pomemben dejavnik pri izvajanju protokolov (lahko imajo omejitve velikosti paketov ali parametrov) ali pri izvajanju na napravah, ki imajo omejen pomnilnik (npr. naprave IoT) in/ali so omejene pri količini prenosa (npr. vgrajene naprave, kjer je potrebno porabiti čim manj energije za prenos podatkov). Velikost deljene skrivnosti (ta vrednost je ob prenosu vsebovana v kriptogramu in zato ne prispeva k velikosti prenesenih podatkov) lahko prav tako vpliva na odločitev o uporabi KEM. Če obstajajo zahteve glede velikosti deljene skrivnosti, ker naj bi se skrivnost uporabila, na primer, kot ključ v AES-256, bi se zahtevala 32 zlogov velika skrivnost. Čeprav se deljena skrivnost praviloma ne uporabi neposredno kot ključ (z varnostnega vidika je lahko to popolnoma sprejemljivo, če uporabljate KEM [39] in ob upoštevanju varnostnih lastnosti vezave), ampak se za pretvorbo skrivnosti v ključe in/ali druge skrivne vrednosti poljubne velikosti uporabi funkcija izpeljave ključa (angl. Key Derivation Function - KDF).

Poleg velikosti elementov algoritmov, Tabela 2 vključuje tudi informacije o varnostnem modelu, po katerem so algoritmi varni, in doseženi NIST varnostni stopnji (več o obeh v nadaljevanju).

Tabela 2: **Osnovne informacije o algoritmih KEM [40].**

KEM z naborom parametrov	Varnostni model	NIST varnostna stopnja	Velikost javnega ključa (zlogi)	Velikost zasebnega ključa (zlogi)	Velikost kriptograma (zlogi)	Velikost deljene skrivnosti (zlogi)
ML-KEM-512	IND-CCA2	1	800	1632 (ali 64 seme)	768	32
ML-KEM-768	IND-CCA2	3	1184	2400 (ali 64 seme)	1088	32
ML-KEM-1024	IND-CCA2	5	1568	3168 (ali 64 seme)	1568	32
Classic-McEliece-348864	IND-CCA2	1	261120	6492	96	32
Classic-McEliece-348864f	IND-CCA2	1	261120	6492	96	32
Classic-McEliece-460896	IND-CCA2	3	524160	13608	156	32
Classic-McEliece-460896f	IND-CCA2	3	524160	13608	156	32
Classic-McEliece-6688128	IND-CCA2	5	1044992	13932	208	32
Classic-McEliece-6688128f	IND-CCA2	5	1044992	13932	208	32
Classic-McEliece-6960119	IND-CCA2	5	1047319	13948	194	32
Classic-McEliece-6960119f	IND-CCA2	5	1047319	13948	194	32
Classic-McEliece-8192128	IND-CCA2	5	1357824	14120	208	32
Classic-McEliece-8192128f	IND-CCA2	5	1357824	14120	208	32
eFrodoKEM-640-AES ali SHAKE	IND-CCA2	1	9616	19888	9720	16
FrodoKEM-640-AES ali SHAKE	IND-CCA2	1	9616	19888	9752	16
eFrodoKEM-976-AES ali SHAKE	IND-CCA2	3	15632	31296	15744	24
FrodoKEM-976-AES ali SHAKE	IND-CCA2	3	15632	31296	15792	24
eFrodoKEM-1344-AES ali SHAKE	IND-CCA2	5	21520	43088	21632	32
FrodoKEM-1344-AES ali SHAKE	IND-CCA2	5	21520	43088	21696	32
BIKE-L1	IND-CPA	1	1541	5223	1573	32
BIKE-L3	IND-CPA	3	3083	10105	3115	32
BIKE-L5	IND-CPA	5	5122	16494	5154	32
HQC-128	IND-CCA2	1	2249	2305	4433	64
HQC-192	IND-CCA2	3	4522	4586	8978	64
HQC-256	IND-CCA2	5	7245	7317	14421	64

3.1 Varnost

V tem razdelku bodo obravnavane različne varnostne lastnosti in varnostni modeli, ki jih je mogoče uporabiti za algoritme KEM (npr. nerazločljivost kriptograma (IND), ki je že omenjena v Tabeli 2. Podrobneje bodo predstavljeni tudi NIST nivoji varnosti, ki so tudi že bili omenjeni v Tabeli 2.

3.1.1 Nerazločljivost kriptograma

Varnostni model nerazločljivosti kriptograma se uporablja za dokazovanje varnosti algoritmov šifriranja. Delujejo kot igra med izzivalcem, ki ima skrivni ključ (tj. zasebni ključ v primeru algoritmov javnega ključa), in nasprotnikom (tj. napadalcem), ki želi razbiti algoritem šifriranja. Del imena »IND« (v IND-CPA ali IND-CCA2) pomeni nerazločljivost kriptograma (angl. Ciphertext Indistinguishability) in se nanaša na nasprotnikovo nezmožnost razlikovanja med šifriranimi sporočili (ali inkapsuliranih skrivnih ključev, v primeru KEM) na podlagi kriptogramov. V igri nasprotnik pošlje dve sporočili v čistopisu (angl. Plaintext) izzivalcu, ki šifrira naključno in vrne kriptogram. Da bi bil algoritem varen (v skladu s tremi modeli, ki jih bomo kmalu predstavili), nasprotnik ne sme imeti bistveno večje možnosti kot 50 % (ker izbira med dvema sporočiloma), da pravilno ugane izvorno sporočilo. Edini način za zmago s kakršno koli doslednostjo je, da izzivalec razbije osnovni matematični problem, na katerem temelji algoritem (npr. problem mreže). Nerazločljivost kriptograma je na voljo v treh nivojih, ki v bistvu določajo, koliko ‚moči' lahko ima napadalec in še vedno ne more

razlikovati med šifriranimi sporočili. Nerazločljivost pri napadu z izbranim čistopisom (angl. Indistinguishability under chosen plaintext attack - IND-CPA) je najšibkejša med njimi.

Z IND-CPA lahko nasprotnik poleg izvajanja poljubnih operacij, šifrira (ker ima dostop do javnega ključa) katerikoli čistopis, ki ga želi in dobi veljaven kriptogram. Tudi potem, ko nasprotnik pošlje dva čistopisa in izzivalec vrne nasprotniku v ugibanje kriptogram enega izmed čistopisov, lahko nasprotnik še vedno šifrira poljubna sporočila. Ideja te igre je dokazati, da napadalec, ki vidi šifrirana sporočila (tj. kriptograme), kar je normalno, iz njih ne more pridobiti nobenih pomembnih informacij. IND-CPA vključuje tudi naključnost šifriranja, sicer bi nasprotnik šifriral čistopisa, poslana izzivalcu, in bi lahko primerjal rezultate z vrnjenim kriptogramom.

Nerazločljivost pri (neprilagodljivem) napadu z izbranim kriptogramom (angl. Indistinguishability under (non-adaptive) chosen ciphertext attack - IND-CCA1) daje nasprotniku vse možnosti iz IND-CPA in doda možnost, da izzivalec dešifrira kriptograme, ki jih je ustvaril. Ko nasprotnik pošlje dva čistopisa izzivalcu, nima več dostopa do možnosti dešifriranja. Nerazločljivost pri napadu s prilagodljivim izbranim kriptogramom (angl. Indistinguishability under adaptive chosen ciphertext attack - IND-CCA2) je enaka kot IND-CCA1 z dodatkom, da lahko nasprotnik dešifrira poljubne kriptograme tudi po prejemu odgovora na izziv (edina izjema je kriptogram vrnjen v izzivu). Ta raven varnosti dokazuje, da tudi če lahko nasprotnik nekoga prelisiči, da dešifrira poljubno število šifriranih sporočil, nasprotniku ne bo dala prednosti in pokaže, da so šifrirana besedila zaščitena pred posegi (sicer bi napadalec lahko predvidljivo spremenil kriptogram izziva in ga dešifriral). V kriptosistemih z IND-CCA2 nivojem varnosti se lahko par ključev obravnava kot dolgoročen in ponovno uporabi [41]. Sicer to na splošno ni najboljša praksa, če se ji je mogoče izogniti. Da bi dosegli popolno prihodnjo varnost (angl. Perfect Forward Secrecy), je treba zasebni ključ izbrisati po dekapsulaciji deljene skrivnosti [42]. Ključe, ki so namenjeni enkratni uporabi imenujemo efemeralni ključi (angl. Ephemeral key). IND-CCA2 vključuje varnostne značilnosti IND-CPA in IND-CCA1, IND-CCA1 pa vključuje lastnosti IND-CPA. [43]

Skoraj vsi post-kvantni algoritmi KEM, pregledani v tem prispevku (glej Tabelo 2) dosežejo varnost IND-CCA2 (pogosto imenovano samo IND-CCA), ki je bila določena tudi v merilih NIST za ocenjevanje varnosti predlaganih rešitev. Rahlo odstopanje je shema BIKE, ki ima dokazano samo varnost IND--CPA; vendar pa bi lahko algoritem dosegel tudi IND-CCA2 (kot je bilo obravnavano v sekciji o algoritmu BIKE).

3.1.2 Varnostne lastnosti vezave

KEM so razmeroma nova oblika algoritmov, vsaj v praksi. Tradicionalno se za izmenjavo ključa uporabljajo konstrukti, ki temeljijo na Diffie-Hellman algoritmu, vendar ti niso primerni za uporabo v post--kvantnem svetu. Zato jih nadomeščamo z algoritmi KEM. Pri prehodu iz tradicionalnih na nove algoritme so bile prenesene tudi varnostne lastnosti (tj. nerazločljivost kriptograma). Novi algoritmi KEM so zato bili zasnovani tako, da izpolnjujejo in dokazujejo le to varnostno lastnost [44]. Na žalost se je izkazalo, da samo ta lastnost morda ne bo dovolj, kot so pokazali napadi s ponovno inkapsulacijo (angl. re--encapsulation attacks).

Napad s ponovno inkapsulacijo [45] izkorišča dejstvo, da je v nekaterih KEM mogoče dekapsulirati kriptogram v deljeno skrivnost/ključ in nato ustvariti drug kriptogram za drug javni ključ, ki se dekapsulira v isto deljeno skrivnost. Posledica tega je, da dve entiteti izračunata isto skupno skrivnost, čeprav ena od njih misli, da komunicira z nasprotnikom. To ni v nasprotju s stopnjo varnosti IND-CCA2.

Raziskovalci [45] so formalizirali nove varnostne lastnosti »vezave« za algoritme KEM. Da bi en kriptografski element (npr. deljena skrivnost/ključ) vezal drugega (npr. javni ključ), mora prvi kriptografski element enolično določiti drugi element (tj. med vrednostmi drugega elementa ni trkov). V primeru napad s ponovno inkapsulacijo mora skrivnost v skupni rabi vezati javni ključ (tj. deljena skrivnost/ključ lahko nastane samo iz določenega javnega ključa) in deljena skrivnost mora vezati kriptogram (tj. dva različna kriptograma se ne moreta dekapsulirati v isto deljeno skrivnost/ključ), da napad ni mogoč. Skupaj obstaja šest veljavnih veznih parov med deljeno skrivnostjo/ključem K, kriptogramom CT in javnim ključem PK. Javni ključ PK ne more vezati kriptograma CT ali deljen skrivnosti K, ker bi to pomenilo, da bi bili kriptogrami in deljene skrivnosti vedno enake. Vezi so lahko tudi med več kriptografskimi elementi. Šest veljavnih vezav vključuje:

- X-BIND-K-CT: Deljena skrivnost/ključ K veže kriptogram CT.
- X-BIND-K-PK: Deljena skrivnost/ključ K veže javni ključ PK.
- X-BIND-CT-K: Kriptogram CT veže deljeno skrivnost/ključ K.
- X-BIND-CT-PK: Kriptogram CT veže javni ključ PK.
- X-BIND-K,CT-PK: Deljena skrivnost/ključ K in kriptogram CT skupaj vežeta javni ključ PK.
- X-BIND-K,PK-CT: Deljena skrivnost/ključ K in javni ključ PK skupaj vežeta kriptogram CT.

Veliko algoritmov KEM implicitno zavrača (angl. implicitly rejecting), kar pomeni, da bo algoritem vedno vrnil vrednost (namesto napake) tudi, ko dekapsulacija ne uspe. To velja za vse algoritme KEM, ki uporabljajo Fujisaki-Okamotovo transformacijo, ki je uporabljena v vseh algoritmih KEM obravnavanih v tem prispevku. Implicitna zavrnitev pomeni, da bo vsako šifrirano besedilo sprejeto, zato vezanje kriptograma na deljeno skrivnost ali javni ključ ni mogoče. To pomeni, da algoritmi z implicitnim zavračanjem ne morejo doseči X-BIND-CT-K ali X-BIND-CT-PK.

X v zapisu je označba mesta za različne varnostne modele nasprotnikov: pošten (angl. Honest - HON), uhajanje (angl. Leak - LEAK) in zlonamerni (angl. Malicious - MAL). V poštenem scenariju pare ključev ustvari funkcija za generiranje ključev KEM. V scenariju uhajanja so ključi tudi pošteno ustvarjeni, nato pa uhajajo nasprotniku (tj. nasprotnik pridobi zasebni ključ in ima možnost dekapsulacije). V zlonamernem scenariju lahko nasprotnik sam ustvari veljavne pare ključev. Višji varnostni model vključuje nižje (npr. varnost MAL-BIND-K-CT vključuje varnost LEAK in HON-BIND-K-CT).

Tabela 3 predstavlja varnostne lastnosti vezav za vseh pet algoritmov KEM obravnavanih v tem članku. Ker vsi implicitno zavračajo algoritme, nobeden od njih ne doseže HON-BIND-CT-K ali HON-BIND--CT-PK (zato te lastnosti niso vključene v tabelo).

Kot je razvidno iz Tabele 3 nobeden od algoritmov ne doseže vseh varnostnih lastnosti vezave. Vendar pa obstajajo nekatere olajševalne okoliščine. Prvič, ni popolnoma jasno, kako pomembne so te varnostne lastnosti za protokole v realni uporabi. Predpostavka za te varnostne lastnosti je, da lahko napadalec manipulira z javnim ključem (ki se vrne skupaj s kriptogramom za dekapsulacijo). To bi povzročilo različne rezultate dekapsulacije in neuspešno izmenjavo. Vendar pa se javni ključi, ki jih posreduje napadalec, običajno ne uporabljajo; namesto tega, ko je za dekapsulacijo potreben javni ključ, se uporabi shranjena vrednost ključa. Predpostavka postane ponovno relevantna, če lahko napadalec nekako manipulira s shranjeno vrednostjo, kar pa bi pomenilo, da ima napadalec že zelo napreden dostop, ki bi ga lahko izkoristil za številne napade. Drugič, obstaja zelo enostavna rešitev za vezanje kriptografskih elementov, ki pa niso dela algoritma KEM. Funkcije izpeljave ključev (KDF) se uporabljajo za razširitev deljene skrivnosti na ustrezno dolžino (npr. HKDF v TLS) in za združevanje več deljenih skrivnosti (npr. v hibridnih KEM). Funkcija KDF se lahko uporabi

Tabela 3: **Varnostne lastnosti vezav za ML-KEM, Classic McEliece, FrodoKEM, BIKE in HQC [46], [47].**

Zavezujoč pojem	ML-KEM	Classic McEliece	FrodoKEM	BIKE	HQC
Hon-Bind-K-PK	✓	×	✓	✓	✓
LEAK-BIND-K-PK	✓	×	✓	×	×
MAL-BIND-K-PK	×	×	×	×	×
Hon-Bind-K, Ct-PK	✓	×	✓	✓	✓
LEAK-BIND-K, CT-PK	✓	×	✓	×	×
MAL-BIND-K, CT-PK	×	×	×	×	×
HON-BIND-K-CT	✓	✓	✓	✓	✓
LEAK-BIND-K-CT	✓	✓	✓	✓	×
MAL-BIND-K-CT	×*	✓	×	✓	×
HON-BIND-K, PK-CT	✓	✓	✓	✓	✓
LEAK-BIND-K, PK-CT	✓	✓	✓	✓	×
MAL-BIND-K, PK-CT	✓	✓	×	✓	×

za povezovanje elementov. Na primer, z dodajanjem javnega ključa kot vhoda v funkcijo KDF, shema postane LEAK-BIND-K-PK, ker je končna deljena skrivnost zdaj vezana na javni ključ. Varnostne lastnosti vezave se lahko dodajo tudi preko protokola, ki obdaja KEM. Alternativno je X-BIND-K-PK mogoče doseči tudi z uporabo robustnega PKE in vključitvijo kriptograma v KDF, ki bo deljeno skrivnost vezal z javnim ključem [47]. To je uporabno tudi zato, ker so javni ključi v post-kvantno varnih algoritmih KEM lahko zelo veliki in posledično njihova vključitev v KDF ni učinkovita.

Če nobeden od teh blažilnih dejavnikov ne zadostuje, je pri uporabi teh algoritmov potrebno biti pozoren, da se ne slepo zanašamo na vsebovanost teh lastnosti. Prav tako se zdi, da ne obstaja noben protokol (v uporabi ali hipotetični protokol), kjer bi izkoriščanje pomanjkljivosti varnostnih lastnosti vezave zlonamernemu napadalcu (MAL) prineslo kritično prednost, ker je napadalec v takšni situaciji (kjer lahko manipulira s pari ključev) preprosto že premočan.

Določene pomanjkljivosti algoritma ML-KEM izhajajo iz uporabljenega načina serializacije (angl. serialization). Čeprav so varnostne lastnosti vezave precej novi, so bili znani v času, ko je NIST standardiziral ML-KEM in rešitev je bila na voljo. Rešitev je bila preprosto shraniti zasebni ključ v obliki semena in ga razširiti vsakič, ko je ključ potreben [48]. NIST je delno prisluhnil in podprl shranjevanje ključev ML-KEM v obliki dveh semen, zaradi česar je ML-KEM MAL-BIND-K-CT-varen (če se uporabljajo semena – zato * v Tabeli 3), vendar ne doseže MAL-BIND-K-PK. To je eden od razlogov, zakaj mnogi raje shranjujejo zasebni ključ kot seme (64 zlogov, ki združuje dve semeni vsako velikosti 32 zlogov). To se odraža tudi v predlaganih implementacijah protokolov z algoritmom ML-KEM, kjer se pogosto uporabljajo semena namesto razširjenih ključev. MAL-BIND-K-PK bi tudi lahko bil dosežen, če bi uporabili samo eno seme, kjer bi izračun napačno oblikovanega zasebnega ključa postal nemogoč, vendar bi to zahtevalo nekaj manjših sprememb algoritma. Zanimivo je, da je bil Kyber, predhodnik ML-KEM, varen za obe varnostni lastnosti, vendar ju je izgubil v preobrazbi v ML-KEM zaradi osredotočanja na učinkovitost algoritma.

3.1.3 NIST varnostne ravni

Tabela 4 vsebuje pet nivojev varnosti, ki jih je definiral NIST [49] za namene natečaja za izbiro post-kvantno varnih algoritmov in ki se danes lahko uporabljajo kot način primerjave varnostne moči posameznih algoritmov oz. njihovih naborov parametrov. Ravni so povezane z varnostjo obstoječih (in dobro preizkušenih) kriptografskih algoritmov. Vseh pet stopenj moči se danes šteje za varnih, vendar se to lahko v prihodnosti spremeni. S povezovanjem ravni varnosti z obstoječimi algoritmi z dobro uveljavljenimi stopnjami varnosti je enostavno prepoznati ravni, če oz. ko postanejo prešibke. Na primer, ko bodo simetrične šifre s 128-bitnimi ključi (npr. AES-128) prešibke za uporabo, to pomeni, da se rešitve PQC z NIST varnostno stopnjo moči 1 ne smejo več uporabljati.

V Tabeli 2 lahko opazite, da algoritmi podpirajo samo NIST varnostne ravni 1, 3 in 5, ki ustrezajo varnostnim nivojem treh velikosti ključev AES, ki so že ustaljeno merilo za raven varnosti. Raven 1 je najboljša možnost za tiste, ki želijo najhitrejšo in najučinkovitejšo rešitev, ki naj bi ostala varna še vsaj nekaj časa. To je optimalna izbira, če je hitrost najpomembnejši izbirni parameter, če imajo naprave omejeno moč, če obstajajo omejitve v količini podatkov, ki jih je mogoče prenašati, ali če je ta raven varnosti dovolj dobra. Raven 5 je popolno nasprotje in daje prednost varnosti pred vsem drugim, predvsem na račun učinkovitosti (časa, moči, pasovne širine in pomnilnika). Raven 3 je zlata sredina in je namenjena za situacije, ko želimo dobra varnostna zagotovila, vendar ni-

Tabela 4: **NIST varnostne ravni [49]**

NIST varnostne ravni	Varnostni opis	Post-kvantni nivo varnosti
1	Vsaj tako težko za razbiti kot AES128 (izčrpno iskanje ključev)	64-bitov
2	Vsaj tako težko za razbiti kot SHA256 (iskanje trkov)	85-bitov
3	Vsaj tako težko za razbiti kot AES192 (izčrpno iskanje ključev)	96-bitov
4	Vsaj tako težko za razbiti kot SHA384 (iskanje trkov)	128-bitov
5	Vsaj tako težko za razbiti kot AES256 (izčrpno iskanje ključev)	128-bitov

smo za to pripravljeni žrtvovati učinkovitosti. Višji nivo varnosti ne zagotavlja le večje odpornosti proti kvantnim računalnikom, temveč daje tudi višja zagotovila pred morebitnimi prihodnjimi ranljivostmi v algoritmih in/ali problemih, na katerih temeljijo (npr. učenje z napakami na mreži). Upoštevajte, da večja varnostna moč (tj. večji parametri) ne nujno izničijo posledic novih ranljivosti.

Post-kvantni nivo varnosti v Tabeli 4 je izračunan na podlagi Groverjevega algoritma [50] (za algoritem AES) in algoritma Brassard et al. [51] (za algoritem SHA). Čeprav 64-bitna varnost (tj. 2^{64} poizkusov za razbitje) danes ne zveni kot dovolj (glede, na to da so minimalni uporabni ključi velikosti 128 bitov), je pomembno vedeti, da Groverjev algoritem ne omogoča vzporednega izvajanja (t.i. paralelizacije) kot jo omogočajo tradicionalni računalniki [52], [53]. Da bi dosegli zmanjšanje računskega časa za S, je potrebnih S^2 kvantnih procesorjev [54]. Na primer, milijarda (10^9) kvantnih računalnikov bi še vedno morala izvesti 2^{49} ponovitev (zmanjšano iz 2^{64}), da bi razbili AES-128. Zato bi morali biti tudi algoritmi na varnostni ravni 1 še kar nekaj časa varni.

Za konec naj omenimo še, da je, ko gledamo na varnost v celotni komunikacijski seji, na splošno dobra ideja ohraniti enako raven varnosti za vse sodelujoče komponente. Utemeljitev je, da združevanje zelo močnih in zelo šibkih komponent ne daje dobrih rezultatov, ker je celoten sistem le tako močan kot njegov najšibkejši člen, in neproporcionalno varnejše komponente le upočasnijo celoten proces. Varnostne ravni vključenih kriptografskih gradnikov bi v idealnem primeru morale biti primerljive. Primer tega je mogoče videti pri TLS, kjer se algoritmi s primerljivimi stopnjami varnosti uporabljajo skupaj (npr. X25519, AES-128 in SHA256, ki imajo vsi 128-bitno varnost za tradicionalne računalnike). Zato nam NIST varnostne ravni služijo tudi kot vodilo za združevanje algoritmov KEM in njihovih naborov parametrov z drugimi kriptografskimi gradniki (npr. simetričnimi šiframi in podpisnimi algoritmi).

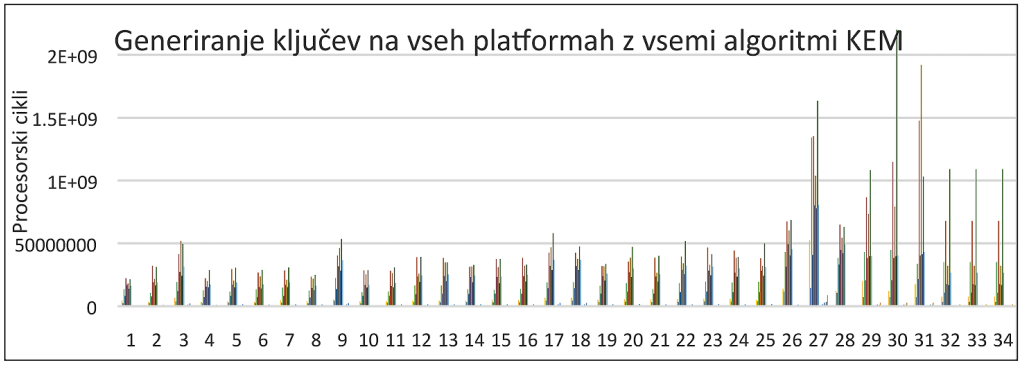
3.2 Hitrost delovanja post-kvantnih algoritmov KEM

To poglavje je namenjeno učinkovitosti delovanja post-kvantnih algoritmov KEM. Zanimala nas bo računalniška zmogljivost generiranja ključev, inkapsulacije in dekapsulacije. Podatki so bili zbrani iz dveh virov: eBATS (ECRYPT Benchmarking of Asymmetric Systems) in podatkov projekta OQS (Open Quantum Safe). V analizo so bili vključeni samo rezultati, ki so bili ustvarjeni za vse opredeljene algoritme na istih platformah.

eBATS [55] je projekt namenjen merjenju učinkovitosti kriptografskih sistemov z javnim ključem. Projekt izgleda, kot da še vedno zbira meritve, vendar algoritmi že nekaj časa niso bili posodobljeni. Za nekatere algoritme KEM je težko oceniti kako stare so že implementacije. Zdi se, da je najstarejša izvedba za BIKE, ki je iz maja 2020 (tik preden je NIST objavil tretji krog tekmovanja) in vključuje samo nabore parametrov L1 in L3 (takrat BIKE ni imel L5) [56]. Glede na opise je uporabljena različica algoritma FrodoKEM-a tudi iz drugega kroga tekmovanja NIST. Čeprav so to nekoliko starejše različice oz. implementacije teh algoritmov, se stanje v novejših različicah ne spreminja drastično, tako da bi se vsaj razmerja med algoritmi moralo ohraniti, čeprav uporabljamo starejše meritve. Ostali algoritmi imajo novejše implementacije (Classic McEliece ima različice pc, f in ne-f, kar nakazuje da gre za trenutne različice algoritmov, HQC je različica 4. kroga, Kyber pa iz 3. kroga, po čemer je bil standardiziran). ML-KEM ni podprt in namesto njega bomo uporabili Kyber. FrodoKEM nima dveh različic, zato bodo rezultati predstavljeni po novem poimenovanju kot eFrodoKEM. eBATS zbira podatke z velikega števila platform[1], vendar smo se omejili samo na tiste, ki so merili najnovejšo različico vseh algoritmov, vključenih v to primerjavo. Čeprav so v eBATS vključene različne platforme, so po odstranitvi procesorjev, ki nimajo meritev za najnovejše izvedbe vseh algoritmov KEM[2], ostale le platforme amd64. Iz eBATS je bilo zbranih podatkov za 25 platform. Rezultati so podani v procesorskih ciklih za generiranje ključev, inkapsulacijo in dekapsulacijo za 25, 50 in 75 percentilov. Uporabili bomo 50 %, kar predstavlja mediano potrebnih ciklov. To je tudi najbolj primerljiva vrednost z OQS rezultati.

Projekt Open Quantum Safe (OQS) [57] je odprtokodni projekt, katerega cilj je podpreti prehod na post-kvantno kriptografijo. Projekt je ustvaril liboqs, ki je odprtokodna knjižnica v programskem jeziku C za kvantno varne kriptografske algoritme. Na žalost njihov primerjalni projekt, ki meri uspešnost njihove knjižnice, ni več aktiven. Njihove zadnje meritve so iz aprila 2024[3]. ML-KEM takrat ni obstajal,

[1] https://bench.cr.yp.to/results-kem.html
[2] https://bench.cr.yp.to/primitives-kem.html
[3] https://openquantumsafe.org/benchmarking/visualization/speed_kem.html



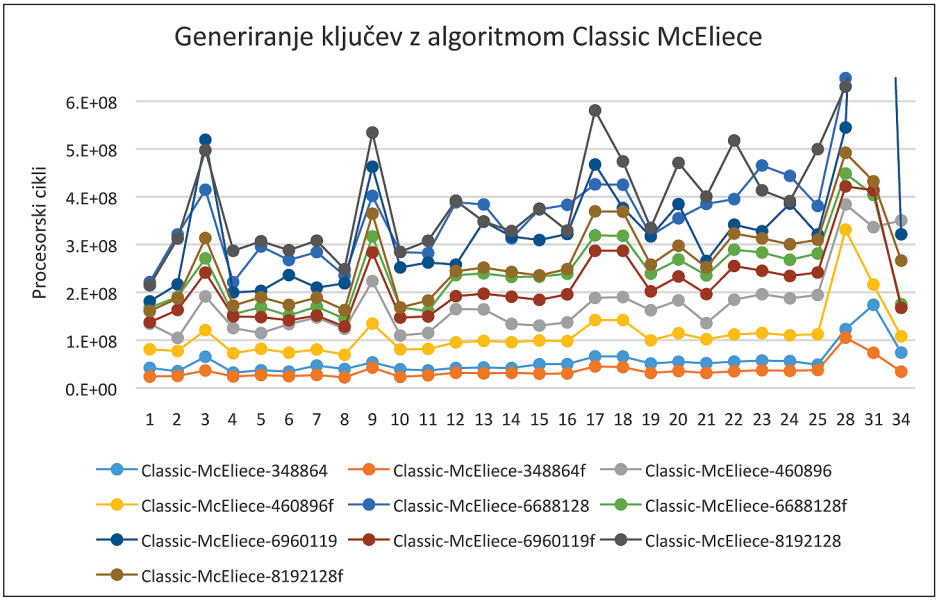
Slika 2: **Hitrost generiranja ključev za vse algoritme KEM na vseh platformah**

zato bomo namesto njega vključili Kyber (čeprav je ML-KEM podprt v novejših različicah knjižnice). Pri prehodu iz Kyberja v ML-KEM je NIST naredil spremembe na algoritmu, zaradi katerih je ML-KEM še hitrejši, kot je bil Kyber. Meritve vključujejo vse obravnavane algoritme KEM, čeprav knjižnica ne podpira obeh različic algoritma FrodoKEM (samo različico iz tretjega kroga NIST tekmovanja). Zato bomo rezultate, poročane za FrodoKEM, spremenili v eFrodoKEM, da bodo odražali spremembo imena. Merila vključujejo tri različne platforme (x86_64, ki je enaka amd64, aarch64 in M1) in tri različne izvedbe algoritmov. Referenčne implementacije so implementirane brez uporabe pospešenih funkcij CPE-ja (Centralno Procesna Enota) in so zato bistveno počasnejše od drugih dveh. Ostali dve implementaciji se razlikujeta glede na to, ali sta prenosni ali ne. To daje devet kombinacij platform in implementacij, vendar nekaterim zaradi njihove neučinkovitosti ne bomo posvečali veliko pozornosti.

Na Sliki 2 je prikazana hitrost generiranja ključev za vse algoritme KEM na vseh 34 platformah (os x), izraženo v procesorskih ciklih (os y). Ta slika ni namenjena predstavi hitrosti delovanja, ampak samo prikazu razlik v delovanju med posameznimi platformami oz. implementacijami. Na sliki lahko hitro vidimo nekaj osamelcev na desni strani grafa. Rezultat označen s številom 27 je referenčna implementacija v OQS, rezultati #29-31 so aarch64 (tj. mobilni procesor), zadnje tri (platforme #32-34) pa so procesor Apple M1. V zadnjih treh kombinacijah M1 platforme in različnih implementacij algoritmov ni razlik, zato v nadaljevanju ne bomo več vključevali vseh treh. Rezultati so podobni, če ne celo slabši, za dekapsulacijo, vendar je presenetljivo, da sta aarch64 in M1 veliko boljša v inkapsulaciji, kjer je M1 najhitrejša platforma, aarch64 pa je prav tako dober, če ne boljši od x86-64. V nadaljevanju bomo izpustili nekatere najslabše rezultate. Specifično so to referenčne implementacije (#27, #30 in #33), ker to niso implementacije, ki bi ji jim bila pomembna hitrost delovanja in prenosne različice implementacij (#26, #29 in #32), ker v povprečju dajejo slabše rezultate kot neprenosne različice.

Hitrost generiranja ključev je na dveh ločenih grafih za Classic McEliece (Slika 3) in ostale algoritme (Slika 4), ker so razlike v zmogljivosti tako velike. Classic McEliece je najpočasnejši algoritem pri generiranju ključev na vseh varnostnih ravneh. Slika 3 prikazuje procesorske cikle, ki so potrebni za generiranje ključev v vseh desetih naborih parametrov Classic McEliece (pet naborov za dve različici z in brez »f« oznake). Hitrost delovanja je zelo konsistentna med nabori parametrov, zlasti v primeru različice f. Rezultati različice, ki ni f imajo visoko varianco na določenih platformah, ki pojasnjuje prepletanje podatkovnih točk na vrhu grafa (trije najpočasnejši nabori parametrov so vse različice, ki niso f in imajo izrazito neoptimalno delovanje na arhitekturi aarch64 - platforma #31). Različice f so tudi konsistentno boljše od različic, ki niso f (pri vseh naborih parametrov). Meritve iz OQS imajo bistveno slabše rezultate za platformo x86_64 (#28) in še posebej za aarch64 (#31).

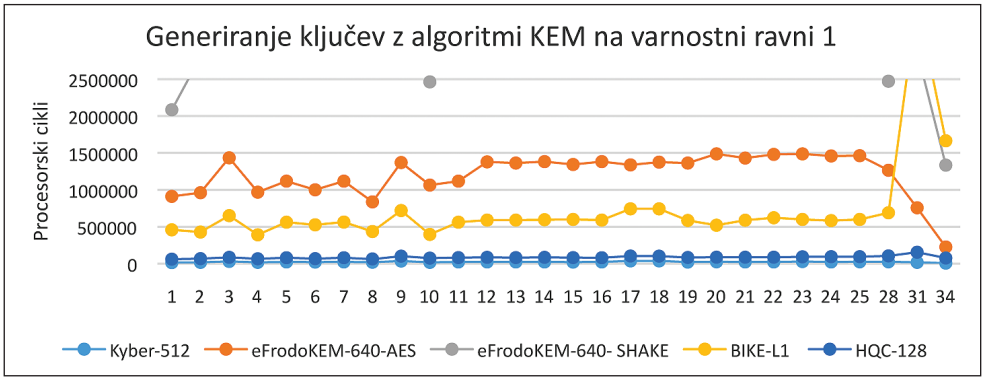
Hitrost generiranja ključev za preostale algoritme KEM z nabori parametrov na varnostni ravni 1 je predstavljena na Sliki 4. Najpočasnejši algoritem (razen Classic McEliece, ki je bil predstavljen ločeno) je FrodoKEM. Implementacija s SHAKE je tako počasna, da je le delno vidna na grafu. Implementacija AES je veliko hitrejša in še posebej učinkovita na platformah aarch64 in M1, kjer algoritem postane skoraj tako hiter kot najuspešnejši algoritmi HQC in Kyber.



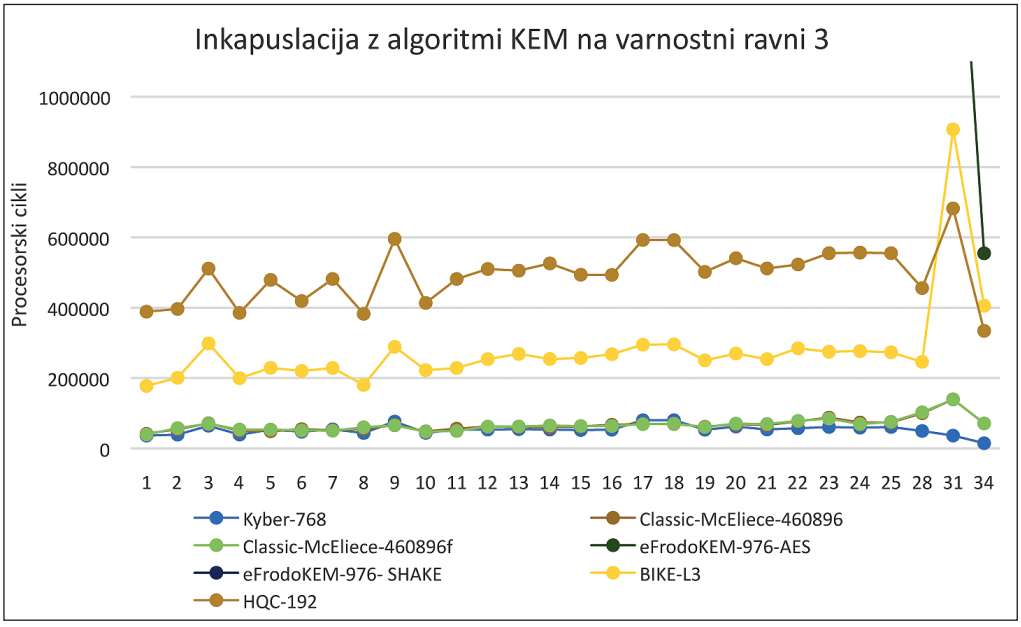
Slika 3: **Hitrost generiranja ključev z algoritmom Classic McEliece.**

BIKE je tretji najhitrejši na x86_64 platformah (#1-28), vendar očitno ni dobro optimiziran za aarch64 ali M1. Kyber je najhitrejši pri generiranju ključev, HQC pa je na drugem mestu. Čeprav Slika 4 predstavlja samo nabore parametrov algoritmov KEM z varnostno ravnjo 1, so razmerja med algoritmi enaka tudi za ravni 3 in 5, kot bo prikazano v tabeli na koncu tega poglavja.

Hitrost inkapsulacije je prikazana na Sliki 5. Na njej lahko vidite število procesorskih ciklov, ki jih vsak algoritem KEM z naborom parametrov na varnostni ravni 3 zahteva za ustvarjanje šifriranega besedila. Najslabši algoritem je eFrodoKEM. eFrodoKEM-SHAKE je tako počasen, da ni viden v grafu. eFrodoKEM-AES je skoraj trikrat hitrejši, vendar še vedno zelo počasen. Na Sliki 5 je eFrodoKEM-AES viden le na skrajni desni strani grafa (platforma #34), kjer optimizacija na platformi M1 omogočajo boljše delovanje. Sledita mu HQC in BIKE, ki imata očitno težave z optimizacijo na mobilnih platformah, sicer pa delujeta šest do dvanajstkrat hitreje kot eFrodoKEM-AES. Dva najuspešnejša algoritma sta Classic McEliece in Kyber. Kyber je le malo hitrejši (in z ML-KEM optimizacijo, bi se morala ta razlika le še okrepiti). Rezultati Classic McEliece različice f in ne-f se bistveno ne razlikujejo. Rezultati oz. razmerja med učinkovitostjo algoritmov se ponovno signifikantno ne razlikujejo, med različnimi nivoji varnosti. Classic McEliece ima več naborov parametrov z varnostno ravnjo 5 in vsi imajo primerljivo hitro delovanje.

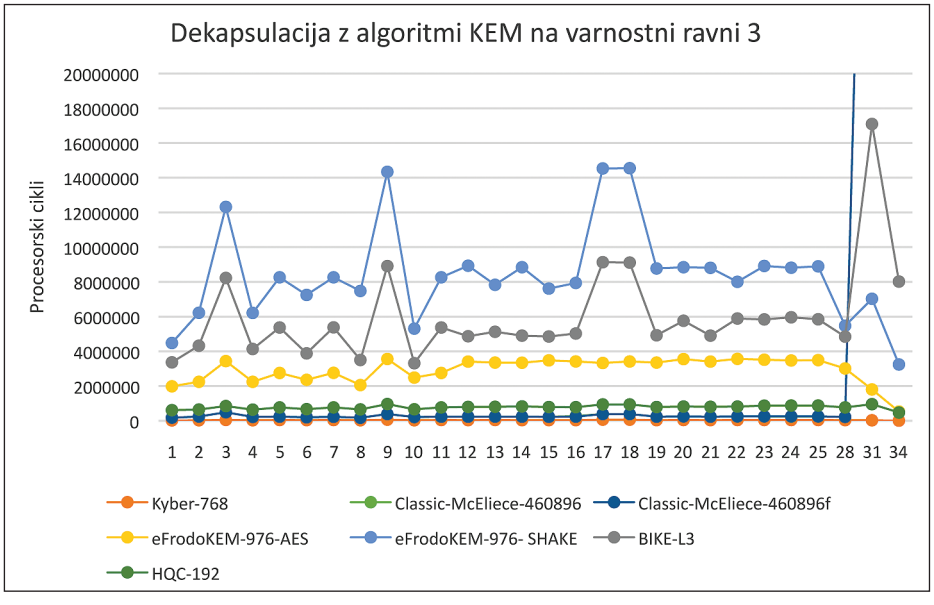


Slika 4: **Hitrost generiranja ključev z algoritmi KEM.**



Slika 5: **Hitrost inkapsulacije z algoritmi KEM.**

Na zadnje imamo na Sliki 6 prikazano še hitrost operacije dekapsulacije. Graf prikazuje povprečno število potrebnih CPE ciklov za dekapsulacijo šifriranega besedila z algoritmi KEM in nabori parametrov za varnostno raven 3. Kot pri ostalih operacijah ostaja razmerje učinkovitosti med algoritmi konsistentno med vsemi tremi nivoji varnosti. eFrodoKEM-SHAKE je spet najslabši. Zanimivo je, da je drugi najpočasnejši BIKE, in šele zatem je eFrodoKEM-AES. Čeprav se zdi, kot da so trije najhitrejši algoritmi za izvajanje dekapsulacije primerljivo hitro so med njimi precej velike razlike. Najučinkovitejši je ponovno Kyber. Classic McEliece je drugi najboljši (različici f in ne-f sta praktično identični), vendar skoraj šestkrat počasnejši od algoritma Kyber (na x86_64 platformah), HQC pa je tretji najhitrejši, a še dodatno trikrat počasnejši od Classic McEliece (na x86_64 platformah). Classic McEliece in BIKE sta bili posebej počasni na



Slika 6: **Hitrost dekapsulacije z algoritmi KEM.**

platformah aarch64 in M1. Možno je, da uporabljeni implementaciji nista bili optimizirani za takšne platforme. Po drugi strani pa je AES očitno zelo dobro optimiziran na teh platformah (zlasti za M1). Classic McEliece nabori parametrov z varnostno ravnjo 5 ponovno dosegajo skoraj identične rezultate.

Na zadnje so tu predstavljene še razlike v zmogljivosti med vsemi algoritmi KEM. Za to primerjavo so algoritmi razdeljeni v kategorije glede na njihovo raven varnosti, tako da se primerjajo samo algoritmi z nabori parametrov na enakem varnostnem nivoju. Rezultati so pokazali, da se implementacije na platformah aarch64 in M1 obnašajo bolj nepredvidljivo. Zato so v končno primerjavo bili vključeni le rezultati platforme x86_64 (tj. platforme #1-26 in #28). Za rezultate algoritma Classic McEliece z varnostno ravnjo 5, so bile uporabljene vrednosti povprečnih procesorskih ciklov vseh treh naborov skupaj, ker delujejo primerljivo hitro. Classic McEliece različici f in non-f sta vztrajno delovali zelo podobno, vendar je bila različica f vedno primerljiva ali boljša. Zato je v končno tabelo rezultatov vključena samo različica f (ker ni razloga, da ne bi uporabili različice f). Enako ne velja za eFrodoKEM, kjer so razlike med različicama AES in SHAKE precejšnje (potencialno bi bil SHAKE veliko bolj primerljiv, če platforme ne bi uporabljale strojno podprtega AES). Zaradi velikih razlik sta v končnih rezultatih različici predstavljeni kot ločena algoritma.

Končna primerjava je predstavljena v Tabeli 5. Tabela je razdeljena na generiranje ključev, inkapsulacijo in dekapsulacijo. Vsaka od treh operacij je nadalje razdeljena na tri varnostne ravni. Za vsakega od algoritmov KEM je poročano povprečno število procesorskih ciklov, potrebnih za izvedbo operacije. Poleg ciklov je delta, ki predstavlja množitelja med številom ciklov algoritma in naslednjim boljšim algoritmom. To pomaga določiti ne le vrstnega reda algoritmov po učinkovitosti, ampak tudi dobro pokazati, kako velike so razlike med njimi. Čeprav se razlike

Tabela 5: **Hitrost algoritmov KEM, ločenih glede na raven varnosti in z delto, ki prikazuje multiplikator do hitrosti prejšnjega algoritma.**

	Generiranje ključev					
	Varnostna raven 1		**Varnostna raven 3**		**Varnostna raven 5**	
KEM	**povp. cikl.**	**Δ**	**Avg. cy.**	**Δ**	**Avg. cy.**	**Δ**
Kyber / ML-KEM	24891	/	42592	/	60343	/
HQC	85176	3,42	200393	4,70	407027	6,75
BIKE	577157	6,78	1633767	8,15	4502751	11,06
eFrodoKEM (AES)	1268476	2,20	2534721	1,55	4286058	0,95
eFrodoKEM (SHAKE)	3842852	3,03	8221838	3,24	14380603	3,36
Classic McEliece (f različice)	37366143	9,72	116149661	14,13	243574748	16,94
	Inkapsulacija					
Kyber / ML-KEM	36033	/	55116	/	78205	/
Classic McEliece (f različice)	29702	0,82	65087	1,18	116681	1,49
BIKE	106749	3,59	249784	3,84	500944	4,29
HQC	216921	2,03	492785	1,97	950774	1,90
eFrodoKEM (AES)	1687015	7,78	3211743	6,52	5333815	5,61
eFrodoKEM (SHAKE)	4149954	2,46	8711655	2,71	15171487	2,84
	Dekapsulacija					
Kyber / ML-KEM	29157	/	45198	/	65767	/
Classic McEliece (f različice)	125742	4,31	261470	5,78	300710	4,57
HQC	378164	3,01	791014	3,03	1560939	5,19
eFrodoKEM (AES)	1615383	4,27	3074454	3,89	5171348	3,31
BIKE	1724035	1,07	5450595	1,77	11126235	2,15
eFrodoKEM (SHAKE)	4080854	2,37	8615275	1,58	14996451	1,35

ne zdijo tako velike, je potrebno upoštevati multiplikativno razmerje med različnimi mesti. Na primer, Kyber-512 (tj. Kyber z varnostno ravnjo 1) je najhitrejši pri generiranju ključev, medtem ko je Classic-McEliece-348864f (na zadnjem mestu) v povprečju 1500-krat počasnejši.

Rezultati za vsako operacijo so med algoritmi KEM zelo konsistentni, z dvema izjemama. BIKE-L5 prehiti eFrodoKEM-AES, pri generiranju ključev na varnostni ravni 5 (razlog je najverjetneje manjše število meritev). Drugi presenetljiv rezultat je hitrejša inkapsulacija Classic-McEliece-348864f kot Kyber-512 na varnostni ravni 1. To je edina varnostna raven, na kateri se to zgodi, in glede na izboljšave ML-KEM v primerjavi s Kyber ni jasno, ali razlika še vedno ostaja.

Končna razvrstitev algoritmov KEM, ki temelji hitrosti njihovega delovanja pri generiranju, inkapsulaciji in dekapsulaciji ključev, je predstavljena v Tabeli 6.

Tabela 6: **Razvrstitev KEM na podlagi učinkovitosti generiranja, inkapsulacije in dekapsulacije ključev.**

#	Generiranje ključev	Inkapsulacija	Dekapsulacija
1	ML-KEM / Kyber	ML-KEM / Kyber	ML-KEM / Kyber
2	HQC	Classic McEliece	Classic McEliece
3	BIKE	BIKE	HQC
4	FrodoKEM (AES)	HQC	FrodoKEM (AES)
5	Classic McEliece	FrodoKEM (AES)	BIKE

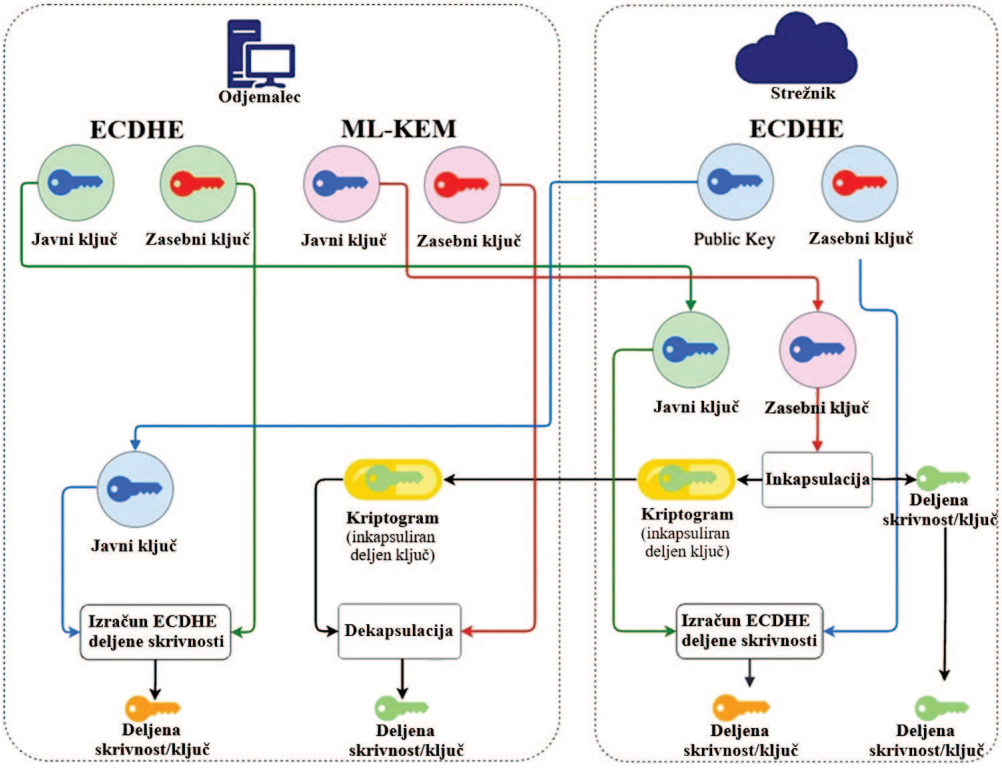
4 UPORABA POST-KVANTNIH ALGORITMOV KEM

Obstaja že kar nekaj (predlaganih) rešitev, ki vključujejo post-kvantne algoritme KEM. Izvajajo se lahko kot samostojni algoritmi KEM ali kot del hibridnega KEM, ki se nato integrira v protokol. Večina rešitev uporablja hibridne KEM rešitve.

Hibridni algoritmi KEM (angl. Hybrid KEM ali PQ/T Hybrid Combiners) so mehanizmi za inkapsulacijo ključa, ki združujejo post-kvanten algoritem KEM s tradicionalnim algoritmom KEM (bolj običajno imenovanim algoritem za izmenjavo ključa ali dogovor o ključu), z namenom doseganja robustnosti (ki jo zagotavljajo dobro uveljavljeni tradicionalni KEM) in odpornost pred kvantnimi računalniki (ki jo zagotavljajo post-kvantni KEM). Poleg varovanja pred morebitnimi neznanimi varnostnimi ranljivostmi in napakami v implementaciji novih post-kvantnih algoritmov KEM, so tradicionalni algoritmi včasih tudi zahtevani zaradi predpisov ali drugih pravil. Ko bodo post-kvantni algoritmi KEM bolje varnostno raziskani in preverjeni čez daljše časovno obdobje, se bo svet lahko vrnil k optimalnejši rešitvi uporabe enega samega algoritma KEM naenkrat. Hibridni algoritem KEM ostane varen, dokler je vsaj en od prispevajočih algoritmov varen. Kot bo razvidno iz primerov v tem poglavju, so hibridni algoritmi KEM le dva (možno je tudi več) neodvisnih KEM-ov, vsak od katerih neodvisno izmenja deljeno skrivnost, ki se nato pretvori v eno skrivnost, ki je odvisna od vseh uporabljenih algoritmov KEM. Razlike med hibridnimi algoritmi KEM izhajajo predvsem iz majhnih razlik v implementaciji (kakšne vrste ključev se uporabljajo ali kako se shranjujejo), uporabljenem naboru parametrov, načinu izračuna končne skupne skrivnosti/ključa in komunikacijskem procesu (tj. način izmenjave sporočil), ki ga opredeljujejo protokoli, v katere so algoritmi KEM integrirani.

Upoštevajte, da v času pisanja tega članka skoraj nobeden od obravnavanih hibridnih algoritmov KEM in njihova vključitev v protokole ni standardizirana. Predstavljeno so večinoma osnutki (angl. Draft) in se bodo v prihodnosti mogoče spremenili in/ali nikoli ne bodo sprejeti.

Najbolj znan in najbolj razširjen primer uporabe hibridnega algoritma KEM je zagotovo v TLS protokolu, ki je bil dobro podprt s strani proizvajalcev brskalnikov in tudi adaptacije s strani večjih spletnih ponudnikov. Post-kvantni hibridni dogovor o ključu ECDHE-MLKEM, razvit za uporabo v TLSv1.3 opredeljuje tri hibridne algoritme KEM [58]. To so X25519MLKEM768, SecP256r1MLKEM768 in SecP384r1MLKEM1024. Prva kombinacija uporablja X25519 ECDH (Elliptic Curve Diffie-Hellman) z ML-KEM in je zamenjava za X25519Kyber768 [59], [60], ki je bil prvi množično uporabljen hibridni KEM, ko je bil med drugim podprt v brskalnikih Chromium, Firefox in Safari ter na strežnikih Googla in Cloudflara [61] za zaščito povezav s TLS 1.3. Kot nakazuje že poimenovanje, ostala druga dva hibridna algoritma KEM združujeta ML-KEM (z naborom parametrov 768 ali 1024) z ECDH secp256r1 (NIST P-256) in secp384r1 (NIST P-384). Vsi omenjeni algoritmi ECDH uporabljajo efemerne ključe (tj. za enkratno uporabo), kar jih naredi ECDHE in predstavljajo glavne tradicionalne algoritme za dogovor o ključu, ki se trenutno uporabljajo. Ločeno je bi predlagan tudi



Slika 7: **Delovanje ECDHE-MLKEM [63]**

curveSM2MLKEM768 [62], ki je zelo enak konstruktom ECDHE-MLKEM, vendar uporablja eliptično krivuljo curveSM2.

V hibridnih algoritmih KEM se tipično kombinira tradicionalni KEM z nižjim nivojem varnosti in post-kvantni KEM z višjim nivojem varnosti (na primer ECDHE X25519 ali secp256r1 s 128-bitno varnostjo, v kombinaciji z ML-KEM-768, ki ima 192-bitno varnost oz. NIST varnostno ravnjo 3). Takšna kombinacija je narejena z namenom zaščite novih in še relativno malo preizkušenih post-kvantnih algoritmov KEM pred napredkom v njihovi kriptoanalizi.

X25519MLKEM768 deluje tako, da odjemalec najprej generira pare ključev za pogajanja (ključe X25519 je treba ustvariti vsakič, ker so efemerni, medtem ko je par ML-KEM-768 mogoče ponovno uporabiti). Oba javna ključa sta poslana strežniku. Strežnik ustvari svoj par ključev X25519 in za inkapsulacijo uporabi odjemalčev javni ključ ML-KEM, iz katerega dobi deljeno skrivnost ML-KEM in kriptogram. Kriptogram in ustvarjeni javni ključ X25519 sta poslana nazaj odjemalcu. Strežnik lahko nato združi odjemalčev javni ključ X25519 in svoj zasebni ključ, da ustvari deljeno skrivnost X25519. Medtem je odjemalec prejel strežnikov javni ključ X25519 in kriptogram iz ML-KEM. Odjemalec dekapsulira prejeti kriptogram s svojim zasebnim ključem ML-KEM, da dobi prvo (ML-KEM) deljeno skrivnosti, nato pa uporabi javni X25519 ključ strežnika in svoj zasebni X25519 ključ, da dobi drugo deljeno skrivnost. Tako odjemalec kot strežnik združita skrivnosti ML-KEM (na prvem mestu) in X25519, ki sta jih pridobila v izmenjavi, da dobita končno deljeno skrivnost. SecP256r1MLKEM768 in SecP384r1MLKEM1024 delujeta na popolnoma enak način, vendar je njihov vrstni red združevanja deljenih skrivnosti obrnjen (X25519MLKEM768 je verjetno edini hibridni KEM, ki ne sledi vrstnemu redu, ki ga nakazuje ime). Konstrukcija končne deljene skrivnosti brez uporabe KDF ali kakršnih koli dodatnih podatkov (npr. obeh javnih ključev ECDHE) razen deljenih skrivnosti je varno le zato, ker se ti hibridni algoritmi KEM uporabljajo v protokolu TLS 1.3, ki že sam razširi deljeno skrivnost/ključ in vključi potrebne dodatne podatke (tj. javne ključe), da se zagotovi IND-CCA2, in predvsem ne-zmožnost preoblikovanja (angl. non-malleability) [59]. Uporaba takšnih skrivnosti ni sicer ni priporočljiva, razen če se ustrezno upravlja, tako kot v TLS 1.3. Obstajajo samozadostni hibridni algoritmi KEM, ki vsebujejo združevanje deljenih skrivnosti,

Tabela 7: **Velikosti ključev, kriptogramov in deljenih skrivnosti/ključev post-kvantnih, tradicionalnih in hibridnih algoritmov KEM (v zlogih).**

Algoritem	Velikost javnega ključa	Velikost zasebnega ključa	Velikost kriptog-rama	Velikost podatkov, ki jih pošlje odjemalec	Velikost podatkov, ki jih pošlje strežnik	Velikost deljene skrivnosti
ML-KEM-768	1184	2400 (64 če je seme)	1088	1184	1088	32
ML-KEM-1024	1568	3168 (64 če je seme)	1568	1568	1568	32
X25519	32	32	32	32	32	32
X448	56	56	56	56	56	56
SecP256r1	65	32	65	65	65	32
SecP384r1	97	48	97	97	97	48
CurveSM2	65	32	65	65	65	32
X25519 MLKEM768	/	/	/	1216	1120	64
SecP256r1 MLKEM768	/	/	/	1249	1153	64
SecP384r1 MLKEM1024	/	/	/	1665	1665	80
CurveSM2MLKEM768	/	/	/	1249	1153	64
MLKEM1024-X448	/	/	/	1624	1624	88

tako da varnost ni odvisna od protokola, ki obdaja KEM. Na Sliki 7, je predstavljena izmenjava podatkov v ECDHE-MLKEM dogovoru za ključ.

Tabela 7 navaja velikosti podatkov post-kvantnih, tradicionalnih in hibridnih algoritmov KEM. V nasprotju s tradicionalnimi izmenjavami ključev vključevanje post-kvantnih algoritmov KEM povzroči signifikantno povečanje velikosti izmenjanih podatkov.

Kako se lahko predlagane hibridne sheme ECDHE-MLKEM ali kateri koli drug KEM vključi v TLS 1.3, je opisano v ločenem dokumentu [64]. Osnutek opredeljuje, kako poteka izmenjava ključev med odjemalci in strežniki, ki podpirajo ali ne podpirajo hibridnih rešitev. Pogajanja in izmenjava potrebnih podatkov (npr. deljenih skrivnosti) se izvaja v okviru TLS 1.3 usklajevanja (angl. Handshake). Velik del tega je opredelitev novih »groups« (TLS poimenovanje za algoritme, uporabljene za izmenjavo skrivnosti/ključa), ki sedaj vključuje možnosti s post-kvantnimi algoritmi KEM. Pristop hibridizacije v TLS je zelo preprost. Informacije o algoritmih, ki sestavljajo hibridno rešitev so poslani v sporočilih usklajevanja in preprosto združeni v eno vrednost, tako da obstoječe podatkovne strukture TLS protokola ostanejo nespremenjene. Končna deljena skrivnost se pridobi tudi z združevanjem rezultatov iz dveh (ali več) uporabljenih KEM-ov (kot je bilo predstavljeno v primeru ECDHE-MLKEM), ki se nato razširi na pričakovano velikost skrivnosti v funkciji izpeljave ključev (tj. KDF). Priporočilo je, da se post-kvantni algoritmi (in tradicionalni) KEM izvajajo z efemernimi ključi. Za post-kvantne algoritme KEM je dovoljena tudi ponovna uporaba parov ključev, vendar le, če to izbrani KEM podpira. Ker ima TLS omejene velikosti javnih ključev in kriptogramov (do 65535 zlogov), Classic McEliece ni primeren za uporabo v protokolu. Google je pripravil tudi osnutek o tem, kako predvidevajo, da bo TLS 1.3 deloval s hibridnimi algoritmi KEM [65], s poudarkom na načinih ublažitve povečane porabe računske zahtevnosti in pasovne širine, ki jo prinaša post-kvantna kriptografija.

X-Wing [66], [67] je hibridni KEM, ki združuje ML-KEM-768 in X25519. X-Wing predvideva uporabo semen namesto razširjenih ML-KEM ključev. To omogoča X-Wingu, da hrani majhno vrednost namesto vseh ključev (kot je bilo obravnavano v razdelku za ML-KEM). Zasebni ključi se lahko začasno shranijo v predpomnilniku (namesto da se shranijo samo kot seme), če se s tem izboljša učinkovitost (npr. v primerih večkratne dekapsulacije z istim ključem ali če se generacija in dekapsulacija ključev zgodita

v kratkem časovnem okviru). Velikost podatkov, ki se prenašajo med dvema odjemalcem in strežnikom je enaka velikosti X25519MLKEM768 (glej Tabelo 7). X-Wing uporablja SHA3-256 kot KDF za izračun končne deljene skrivnosti z zgoščevanjem deljene skrivnosti ML-KEM-768, deljene skrivnosti X25519, X25519 kriptograma, obeh javnih ključev X25519 in konstante (imenovane XWingLabel). X-Wing je bil narejen kot samostojni hibridni algoritem KEM, kar pomeni, da ima dokazano celovito varnost in ga je mogoče vstaviti kot zamenjavo za dogovor o ključu v katerem koli protokolu (npr. TLS, HPKE, SSH itd.) brez sprememb ali dodatnega upravljanja.

X-Wing določa kombinacijo algoritma ECDHE (X25519) s post-kvantnim ML-KEM, vendar bi lahko hibridni algoritem KEM zgradili na enak način iz poljubne kombinacije tradicionalnega in post-kvantnega algoritma KEM (ob upoštevanju nastalih varnostnih lastnosti). V ta namen je bil pripravljen osnutek za generično ustvarjanje hibridnih algoritmov KEM [68]. Opredeljuje generične tehnike za doseganje hibridnih KEM iz poljubnih post-kvantnih in tradicionalnih algoritmov KEM, ki ustrezajo specifičnim varnostnim lastnostim. Osnutek predlaga dve različni konstrukciji: KitchenSink in QSF. KitchenSink je bolj splošna možnost, zasnovana tako, da lahko združi največje število tradicionalnih in post-kvantnih algoritmov KEM, medtem ko je QSF bolj prilagojen in zahteva KEM s specifičnimi lastnostmi. X-Wing je v bistvu specifična različica QSF oblike izgradnje hibridnega algoritma KEM. Obstajajo tudi generični kombinatorji, kjer je pomembno, samo da imajo kombinirane sheme neodvisno generiranje ključev, način združevanja deljenih skrivnosti pa je pomemben za varnost celotnega sistema. Odobreni načini za združevanje deljenih skrivnosti dveh (ali več) KEM-ov so opredeljeni v [69] in [70].

Nazadnje je tu omenjenih še nekaj drugih protokolov, ki že načrtujejo oz. mogoče tudi že delujejo z uporabo hibridnih algoritmov KEM. Vsi hibridni algoritmi KEM so ustvarjeni na zelo podobne ali identične načine opisanemu. Tipično edine večje razlike so v tem kako se ti algoritmi integrirajo v protokole ali uporaba KDF. Za Internet Key Exchange Protocol Version 2 (IKEv2) so bilo pripravljenih precej osnutkov v pripravi na vključitev post-kvantnih in hibridnih algoritmov KEM [71], [72] vključno s predlogi o vključevanju specifičnih algoritmov [73], [74]. Nekateri ponudniki VPN so že vključili post-kvantne algoritme KEM v svoje protokole. ExpressVPN je integriral ML-KEM (in pred tem Kyber) v svoj protokol Lightway, NordVPN je integriral ML-KEM v svoj protokol NordLynx [75], in obstaja že predlog za post-kvantni WireGuard [76]. Cryptographic Message Syntax (CMS) v RFC 5652 [77] že predvideva uporabo post-kvantnih algoritmov KEM. Hibridne rešitve za CMS in X.509 PKI so bile predlagane v ločenem osnutku [78]. Predlog za uporabo post-kvantne kriptografije v OpenPGP vključuje hibridne KEM in hibridne sheme podpisov javnih ključev [79]. Signal je predstavil PQXDH [80], ki je post-kvantno varna zamenjava za njihov 3XDH protokol. Za razliko od ostalih protokolov, omenjenih v tem poglavju je PQXDH namenjen asinhronemu delovanju. Še ena post-kvantno varna alternativa protokolu 3XDH je K-Waay [81].

5 SKLEP

Razvoj kvantnih računalnikov se neomajno nadaljuje in njihov napredek je začel ogrožati kriptografske elemente, na katere se danes zanašamo za overjanje in varovanje komunikacije ter podatkov. Kvantni računalniki še niso dovolj zmogljivi, da bi ogrozili varnost trenutne kriptografije, vendar je njihov napredek dovolj hiter, da predstavljajo realno grožnjo za podatke, ki so danes zavarovani, vendar bodo še vedno imeli vrednost tudi v času, ko bodo kvantni računalniki postali dovolj zmogljivi, da bodo lahko razkrili te podatke. V preprečevanje tega, je potrebno čimprej nadomestiti uporabo ranljive kriptografije s kvantno-varnimi alternativami. Prispevek predstavlja širok pregled post-kvantnih mehanizmov za inkapsulacijo ključev (KEM), ki so pomemben del post-kvantne kriptografije. Algoritmi KEM skrbijo za varno izmenjavo skrivnosti na podlagi katerih se varuje nadaljnja komunikacija.

V prispevku smo se omejili na post-kvantno kriptografijo namenjeno izmenjavi ključev, zato algoritmi namenjeni overjanju oz. podpisovanju niso bili vključeni. Pri pregledu specifičnih post-kvantnih KEM algoritmov smo se omejili na algoritme, ki so se uvrstili v zadnji krog NIST-ovega tekmovanja za standardizacijo post-kvantnih algoritmov, algoritme, ki so v tekmovanju že bili izbrani za standardizacijo in algoritem FrodoKEM, ker ga nekatere pomembne organizacije izpostavljajo kot dober rezerven algoritem. Skupaj je bilo vključenih pet algoritmom KEM, za katere je bila predstavljena njihova osnovna konstrukcija, nabori

parametrov, varnostne lastnosti in učinkovitost njihovega delovanja. Pri analizi učinkovitosti oz. hitrosti njihovega delovanja smo se omejili na že zbrane in javno dostopne podatke.

Če povzamemo varnostne lastnosti kvantno varnih algoritmov KEM, razlike med njimi niso velike. Večinoma vsi dosegajo varnost IND-CCA2, čeprav je ob uporabi kratkotrajnih ključev zadostna IND-CPA. Lastnosti vezave so med algoritmi bolj različne, čeprav je glede na novost teh lastnosti možno, da bo še prišlo do sprememb (predvsem nestandardiziranih) algoritmov, ki bi spremenile varnostne lastnosti vezave. Ne glede na to, pomanjkanje nekaterih od teh lastnosti ni nujno velika ovira za KEM, vendar je pomembno te pomanjkljivosti upoštevati v KDF ali samem protokolu.

Ob prehodu na uporabo algoritmov KEM obstaja velika verjetnost, da bo shranjevanje semen prednostna oblika pred shranjevanjem razširjenih zasebnih ključev (vsaj v primeru ML-KEM). Izvajanje algoritmov z uporabo semen ponuja nekatere varnostne prednosti (npr. MAL-BIND-K-CT), porabi manj prostora za shranjevanje in naj bi bilo manj nagnjeno k napakam pri implementaciji (semena za razliko od dekapsulacijskega ključ ni potrebno vedno preverjati).

NIST varnostna raven 1 trenutno smatra kot popolnoma varna, čeprav je pomembno podpirati tudi višje ravni (če bo nekoč potrebno preiti na te). Dejansko izbiro ravni je potrebno opraviti glede na okoliščine uporabe in v skladu z drugimi kriptografskimi gradniki. Okoliščine, med drugim, vključujejo napredek kvantnih računalnikov, namen in okolje izvajanja. Ker s časom kvantni računalniki napredujejo, in če pride do pomembnih prebojev, ki znatno povečajo njihov napredek/moč, ali če se varnostne zahteve ali pomen zaščitenih podatkov sčasoma spreminja, je najboljša možnost spremembe varnostne ravni brez spreminjanja implementacije. To je povezano s kripto agilnostjo, ki je lastnost sistema, da lahko spreminja kriptografske mehanizme kot odziv na spreminjajoče se okoliščine. To vključuje nabore parametrov in/ali velikosti ključev ter tudi možnost spreminjanja algoritmov ali knjižnic. Kripto agilnost je strategija, ki zagotavlja trdno in odporno kibernetsko varnost.

Razpored algoritmov KEM glede na njihovo hitrost oz. učinkovitost delovanja ni takoj očiten. Najuspešnejši algoritem je zagotovo ML-KEM (prej imenovan Kyber). Drugi najhitrejši algoritem je težje določiti. Classic McEliece zavzema drugo mesto, če je namen ponovna uporaba ključev in posledično le redko generiranje novega para ključev. Če je pričakovana/zahtevana uporaba kratkotrajnih ključev, potem Classic McEliece ni učinkovita rešitev in je druga najučinkovitejša rešitev HQC. Glede na to da je potrebna uporaba efemernih ključev za zagotavljanje popolne prihodnje varnosti, je ponovna uporaba para ključev redka. To je predvidoma tudi razlog zakaj je NIST izbral HQC kot alternativni algoritem za standardizacijo in uporabo poleg ML-KEM. Naslednji najuspešnejši algoritem je BIKE, ki je v določenih operacijah celo boljši kot HQC. FrodoKEM je konsistentno počasnejši od ostalih algoritmov. Medtem ko različice, ki uporabljajo algoritem AES še lahko proizvedejo dobre rezultate (zlasti na strojni opremi, ki lahko v celoti izkoristi strojno podporo algoritma AES), različice SHAKE sploh niso konkurenčne. Morda bi bili rezultati te različice boljši na specializirani strojni opremi (čeprav to lahko velja za vse algoritme KEM). Tu je potrebno še upoštevati, da je ta razporeditev pripravljena zgolj na podlagi učinkovitosti delovanja. Varnost in velikost ključev ter kriptogramov KEM bi prav tako morali igrati vlogo pri izbiri najboljšega algoritma KEM za določen namen, čeprav glede na vse rezultate predstavljene v tem prispevku je ML-KEM praktično vedno najboljša izbira.

V zadnjem delu prispevka so bile predstavljene obstoječe aplikacije in/ali predloge za vključitev post-kvantnih algoritmov KEM v protokole ali hibridne algoritme KEM. Slednji je kombinacija tradicionalnega in post-kvantnega algoritma KEM, ki bo postal prehodni korak med tradicionalnimi dogovori o ključu in bodočo izključno uporabo post-kvantne kriptografije. Podrobneje predstavimo predvsem primer uporabe hibridnega algoritma KEM v protokolu TLS 1.3, ker ta predstavlja najbolj razširjen primer uporabe takšnih algoritmov. Zasnova predlaganih post-kvantnih protokolov odraža in poudarja specifične lastnosti algoritmov, obravnavane v prejšnjih delih prispevka.

Afiliacije

Ta raziskava je nastala kot rezultat projekta PQC Key Manager, ki ga je financirala Evropska vesoljska agencija (pog. št. 4000146856/24/NL/MH/mp) ter ob podpori raziskovalnega programa št. P2-0057, ki ga je sofinancirala Javna agencija za znanstvenoraziskovalno in inovacijsko dejavnost Republike Slovenije iz državnega proračuna.

LITERATURA

[1] P. W. Shor, »Algorithms for quantum computation: Discrete logarithms and factoring«, *Proceedings - Annual IEEE Symposium on Foundations of Computer Science, FOCS*, str. 124–134, 1994, doi: 10.1109/SFCS.1994.365700.

[2] D. Moody, R. Perlner, A. Regenscheid, A. Robinson, in D. Cooper, »Transition to Post-Quantum Cryptography Standards«, nov. 2024. doi: 10.6028/NIST.IR.8547.IPD.

[3] »Securing Tomorrow, Today: Transitioning to Post-Quantum Cryptography«, 27. november 2024, *BSI - Bundesamt für Sicherheit in der Informationstechnik*. Pridobljeno: 4. april 2025. [Na spletu]. Dostopno na: https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Crypto/PQC-joint-statement.pdf?__blob=publicationFile&v=3

[4] NIST, »Post-Quantum Cryptography Standardization«. Pridobljeno: 3. februar 2025. [Na spletu]. Dostopno na: https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization

[5] C. Gidney in M. Ekerå, »How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits«, *Quantum*, let. 5, str. 433, apr. 2021, doi: 10.22331/q-2021-04-15-433.

[6] »Discover the World's Largest Quantum Computer in 2025«, spinquanta.com. Pridobljeno: 3. februar 2025. [Na spletu]. Dostopno na: https://www.spinquanta.com/newsDetail/ab2b9158-0907-4028-95b7-f8c5efdd9a2b

[7] S. Turner, P. Kampanakis, J. Massimo, in B. Westerbaan, »Internet X.509 Public Key Infrastructure - Algorithm Identifiers for the Module-Lattice-Based Key-Encapsulation Mechanism (ML-KEM) - draft-ietf-lamps-kyber-certificates-08«, feb. 2025. Pridobljeno: 3. februar 2025. [Na spletu]. Dostopno na: https://datatracker.ietf.org/doc/draft-ietf-lamps-kyber-certificates/

[8] J. Massimo, P. Kampanakis, S. Turner, in B. Westerbaan, »Internet X.509 Public Key Infrastructure: Algorithm Identifiers for ML-DSA - draft-ietf-lamps-dilithium-certificates-07«, feb. 2025. Pridobljeno: 3. februar 2025. [Na spletu]. Dostopno na: https://datatracker.ietf.org/doc/draft-ietf-lamps-dilithium-certificates/

[9] M. Ounsworth, J. Gray, M. Pala, J. Klaußner, in S. Fluhrer, »Composite ML-DSA For use in X.509 Public Key Infrastructure and CMS - draft-ietf-lamps-pq-composite-sigs-03«, okt. 2024. Pridobljeno: 3. februar 2025. [Na spletu]. Dostopno na: https://datatracker.ietf.org/doc/draft-ietf-lamps-pq-composite-sigs/

[10] NIST, »Announcing PQC Candidates to be Standardized, Plus Fourth Round Candidates«. Pridobljeno: 3. februar 2025. [Na spletu]. Dostopno na: https://csrc.nist.gov/news/2022/pqc-candidates-to-be-standardized-and-round-4

[11] »NIST Selects HQC as Fifth Algorithm for Post-Quantum Encryption«, NIST. Pridobljeno: 8. april 2025. [Na spletu]. Dostopno na: https://www.nist.gov/news-events/news/2025/03/nist-selects-hqc-fifth-algorithm-post-quantum-encryption

[12] BSI, »Migration to Post Quantum Cryptography - Recommendations for action by the BSI«, 2021. Pridobljeno: 31. januar 2025. [Na spletu]. Dostopno na: https://www.bsi.bund.de

[13] »Follow up position paper on Post-Quantum Cryptography«, okt. 2023. Pridobljeno: 11. februar 2025. [Na spletu]. Dostopno na: https://cyber.gouv.fr/en/publications/follow-position-paper-post-quantum-cryptography

[14] NIST, »FIPS 203, Module-Lattice-Based Key-Encapsulation Mechanism Standard«, avg. 2024, doi: 10.6028/NIST.FIPS.203.

[15] C. Aguilar Melchor *idr.*, »HammingQuasi-Cyclic (HQC) - Fourth round version«, okt. 2024, Pridobljeno: 4. februar 2025. [Na spletu]. Dostopno na: https://pqc-hqc.org/documentation.html

[16] »Classic McEliece: Specification«. Pridobljeno: 4. februar 2025. [Na spletu]. Dostopno na: https://classic.mceliece.org/spec.html

[17] »FrodoKEM: Learning With Errors Key Encapsulation«, dec. 2024, Pridobljeno: 4. februar 2025. [Na spletu]. Dostopno na: https://frodokem.org/#spec

[18] N. Aragon *idr.*, »BIKE: Bit Flipping Key Encapsulation - Round 4 Submission«, okt. 2024, Pridobljeno: 4. februar 2025. [Na spletu]. Dostopno na: https://bikesuite.org/#spec

[19] M. Harmalkar, K. Jain, in P. Krishnan, »A Survey of Post Quantum Key Encapsulation Mechanism«, *Proceedings - 2024 5th International Conference on Mobile Computing and Sustainable Informatics, ICMCSI 2024*, str. 141–149, 2024, doi: 10.1109/ICMCSI61536.2024.00028.

[20] A. Langlois in D. Stehlé, »Worst-case to average-case reductions for module lattices«, *Des Codes Cryptogr*, let. 75, št. 3, str. 565–599, jun. 2015, doi: 10.1007/S10623-014-9938-4/METRICS.

[21] E. Fujisaki in T. Okamoto, »Secure integration of asymmetric and symmetric encryption schemes«, *Journal of Cryptology*, let. 26, št. 1, str. 80–101, jan. 2013, doi: 10.1007/S00145-011-9114-1/METRICS.

[22] J. B. Almeida *idr.*, »Formally verifying Kyber Episode V: Machine-checked IND-CCA security and correctness of ML-KEM in EasyCrypt«, *Cryptology ePrint Archive*, 2024, Pridobljeno: 3. februar 2025. [Na spletu]. Dostopno na: https://eprint.iacr.org/2024/843

[23] F. Valsorda, »Let's All Agree to Use Seeds as ML-KEM Keys«. Pridobljeno: 10. februar 2025. [Na spletu]. Dostopno na: https://words.filippo.io/dispatches/ml-kem-seeds/

[24] H. Singh, »Code based Cryptography: Classic McEliece«, jul. 2019, Pridobljeno: 4. februar 2025. [Na spletu]. Dostopno na: https://arxiv.org/abs/1907.12754v2

[25] R. J. McEliece, »A public-key cryptosystem based on algebraic coding theory«, *Prog. Rep., Jet Prop. Lab., California Inst. Technol., Pasadena, CA, 114–116*, 1978.

[26] M. Á. González de la Torre in L. Hernández Encinas, »About the Fujisaki-Okamoto Transformation in the Code-Based Algorithms of the NIST Post-quantum Call«, *Lecture Notes in Networks and Systems*, let. 532 LNNS, str. 75–85, 2023, doi: 10.1007/978-3-031-18409-3_8.

[27] »Classic McEliece: conservative code-based cryptography: design rationale«, okt. 2022. Pridobljeno: 4. februar 2025. [Na spletu]. Dostopno na: https://classic.mceliece.org/mceliece-rationale-20221023.pdf

[28] »BSI TR-02102-1: Cryptographic Mechanisms: Recommendations and Key Lengths«, jan. 2024. Pridobljeno: 11. februar 2025. [Na spletu]. Dostopno na: https://www.bsi.bund.de/EN/Themen/Unternehmen-und-Organisationen/Standards-und-Zertifizierung/Technische-Richtlinien/TR-nach-Thema-sortiert/tr02102/tr02102_node.html

[29] »Classic McEliece: ISO«. Pridobljeno: 11. februar 2025. [Na spletu]. Dostopno na: https://classic.mceliece.org/iso.html

[30] »Classic McEliece: conservative code-based cryptography: what plaintext confirmation means«, okt. 2023. Pridobljeno: 4. februar 2025. [Na spletu]. Dostopno na: https://classic.mceliece.org/nist.html

[31] »Classic McEliece: conservative code-based cryptography: guide for implementors«, okt. 2022, Pridobljeno: 4. februar 2025. [Na spletu]. Dostopno na: https://classic.mceliece.org/nist.html

[32] »FrodoKEM Learning With Errors Key Encapsulation - Annex on FrodoKEM updates«. Pridobljeno: 5. februar 2025. [Na

spletu]. Dostopno na: https://frodokem.org/files/FrodoKEM--annex-20230418.pdf
[33] NIST, »Advanced Encryption Standard (AES) - FIPS 197«, maj 2023, doi: 10.6028/NIST.FIPS.197-UPD1.
[34] NIST, »SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions - FIPS 202«, avg. 2015, doi: 10.6028/NIST.FIPS.202.
[35] »FrodoKEM«. Pridobljeno: 5. februar 2025. [Na spletu]. Dostopno na: https://frodokem.org/
[36] M. Á. González de la Torre in L. Hernández Encinas, »About the Fujisaki-Okamoto Transformation in the Code-Based Algorithms of the NIST Post-quantum Call«, *Lecture Notes in Networks and Systems*, let. 532 LNNS, str. 75–85, 2023, doi: 10.1007/978-3-031-18409-3_8.
[37] S. Arpin, J. B. Lau, R. Perlner, A. Robinson, J.-P. Tillich, in V. Vasseur, »Error floor prediction with Markov models for QC--MDPC codes«, *Cryptology ePrint Archive*, 2025, Pridobljeno: 5. februar 2025. [Na spletu]. Dostopno na: https://eprint.iacr.org/2025/153
[38] D. Hofheinz, K. Hövelmanns, in E. Kiltz, »A Modular Analysis of the Fujisaki-Okamoto Transformation«, *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, let. 10677 LNCS, str. 341–371, 2017, doi: 10.1007/978-3-319-70500-2_12/FIGURES/19.
[39] G. Alagic *idr.*, »Recommendations for Key-Encapsulation Mechanisms - NIST SP 800-227«, jan. 2025, doi: 10.6028/NIST.SP.800-227.IPD.
[40] »Open Quantum Safe - Algorithms«. Pridobljeno: 22. januar 2025. [Na spletu]. Dostopno na: https://openquantumsafe.org/liboqs/algorithms/
[41] D. Connolly, »ML-KEM Post-Quantum Key Agreement for TLS 1.3 - draft-connolly-tls-mlkem-key-agreement-05«. Pridobljeno: 5. februar 2025. [Na spletu]. Dostopno na: https://datatracker.ietf.org/doc/draft-connolly-tls-mlkem-key-agreement/
[42] S. Fluhrer, Q. Dang, J. Preuß Mattsson, K. Milner, in D. Shiu, »ML-KEM Security Considerations - draft-sfluhrer-cfrg-ml--kem-security-considerations-02 «, 2024. Pridobljeno: 3. februar 2025. [Na spletu]. Dostopno na: https://datatracker.ietf.org/doc/draft-sfluhrer-cfrg-ml-kem-security-considerations/
[43] R. Cramer in V. Shoup, »Design and Analysis of Practical Public-Key Encryption Schemes Secure against Adaptive Chosen Ciphertext Attack«, *Cryptology ePrint Archive*, 2001, Pridobljeno: 5. februar 2025. [Na spletu]. Dostopno na: https://eprint.iacr.org/2001/108
[44] D. Connolly, »How to Hold KEMs«. Pridobljeno: 11. februar 2025. [Na spletu]. Dostopno na: https://durumcrustulum.com/2024/02/24/how-to-hold-kems/
[45] C. Cremers, A. Dax, in N. Medinger, »Keeping Up with the KEMs: Stronger Security Notions for KEMs and Automated Analysis of KEM-based Protocols«, *CCS 2024 - Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security*, str. 1046–1060, dec. 2024, doi: 10.1145/3658644.3670283.
[46] J. Krämer, P. Struck, in M. Weishäupl, »Binding Security of Implicitly-Rejecting KEMs and Application to BIKE and HQC«, *Cryptology ePrint Archive*, 2024, Pridobljeno: 11. februar 2025. [Na spletu]. Dostopno na: https://eprint.iacr.org/2024/1233
[47] C. Cremers, A. Dax, in N. Medinger, »Keeping Up with the KEMs: Stronger Security Notions for KEMs and automated analysis of KEM-based protocols«, *Cryptology ePrint Archive*, nov. 2024, Pridobljeno: 12. februar 2025. [Na spletu]. Dostopno na: https://eprint.iacr.org/2023/1933
[48] S. Schmieg, »Unbindable Kemmy Schmidt: ML-KEM is neither MAL-BIND-K-CT nor MAL-BIND-K-PK«, *Cryptology ePrint Archive*, apr. 2024, Pridobljeno: 11. februar 2025. [Na spletu]. Dostopno na: https://eprint.iacr.org/2024/523
[49] NIST, »Post-Quantum Cryptography - Security (Evaluation Criteria)«. Pridobljeno: 5. februar 2025. [Na spletu]. Dostopno na: https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/evaluation-criteria/security-(evaluation-criteria)
[50] L. K. Grover, »A fast quantum mechanical algorithm for database search«, *Proceedings of the Annual ACM Symposium on Theory of Computing*, let. Part F129452, str. 212–219, jul. 1996, doi: 10.1145/237814.237866/ASSET/0A4D7788-347B-41F8-9E19-26B6DE2ACF0A/ASSETS/237814.237866.FP.PNG.
[51] G. Brassard, P. Hoyer, in A. Tapp, »Quantum Algorithm for the Collision Problem«, *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, let. 1380, str. 163–169, maj 1997, doi: 10.1007/BFb0054319.
[52] C. Zalka, »Grover's quantum searching algorithm is optimal«, *Phys Rev A (Coll Park)*, let. 60, št. 4, str. 2746, okt. 1999, doi: 10.1103/PhysRevA.60.2746.
[53] S. Fluhrer, »Reassessing Grover's Algorithm«, *Cryptology ePrint Archive*, 2017, Pridobljeno: 5. februar 2025. [Na spletu]. Dostopno na: https://eprint.iacr.org/2017/811
[54] Sarah D., »On the Practical cost of Grover for AES Key Recovery«, v *Fifth PQC Standardization Conference*, apr. 2024. Pridobljeno: 6. februar 2025. [Na spletu]. Dostopno na: https://csrc.nist.gov/Presentations/2024/practical-cost-of--grover-for-aes-key-recovery
[55] »eBATS: ECRYPT Benchmarking of Asymmetric Systems«. Pridobljeno: 13. februar 2025. [Na spletu]. Dostopno na: https://bench.cr.yp.to/ebats.html
[56] »List of key-encapsulation mechanisms measured«. Pridobljeno: 13. februar 2025. [Na spletu]. Dostopno na: https://bench.cr.yp.to/primitives-kem.html
[57] »Open Quantum Safe«. Pridobljeno: 13. februar 2025. [Na spletu]. Dostopno na: https://openquantumsafe.org/
[58] K. Kwiatkowski, P. Kampanakis, B. Westerbaan, in D. Stebila, »Post-quantum hybrid ECDHE-MLKEM Key Agreement for TLSv1.3 - draft-kwiatkowski-tls-ecdhe-mlkem-03«. Pridobljeno: 10. februar 2025. [Na spletu]. Dostopno na: https://datatracker.ietf.org/doc/draft-kwiatkowski-tls-ecdhe-mlkem/
[59] B. Westerbaan in D. Stebila, »X25519Kyber768Draft00 hybrid post-quantum key agreement - draft-tls-westerbaan-xyber768d00-03«. Pridobljeno: 10. februar 2025. [Na spletu]. Dostopno na: https://datatracker.ietf.org/doc/draft-tls-westerbaan-xyber768d00/
[60] J. Schaumann, »TLS 1.3 Hybrid Key Exchange using X25519Kyber768 / ML-KEM«. Pridobljeno: 10. februar 2025. [Na spletu]. Dostopno na: https://www.netmeister.org/blog/tls-hybrid-kex.html
[61] »Post-Quantum Key Agreement at Cloudflare«. Pridobljeno: 10. februar 2025. [Na spletu]. Dostopno na: https://pq.cloudflareresearch.com/
[62] P. Yang, C. Peng, J. Hu, in S. Sun, »Hybrid Post-quantum Key Exchange SM2-MLKEM for TLSv1.3 - draft-yang-tls--hybrid-sm2-mlkem-01«. Pridobljeno: 10. februar 2025. [Na spletu]. Dostopno na: https://datatracker.ietf.org/doc/draft--yang-tls-hybrid-sm2-mlkem/
[63] U. Pathum, »X25519Kyber768 Post-Quantum Key Exchange for HTTPS Communication«, Medium. Pridobljeno: 10. februar 2025. [Na spletu]. Dostopno na: https://medium.com/@

hwupathum/x25519kyber768-post-quantum-key-exchange-for-https-communication-70eba681931d
[64] D. Stebila, S. Fluhrer, in S. Gueron, »Hybrid key exchange in TLS 1.3 - draft-ietf-tls-hybrid-design-12«, Internet Engineering Task Force (IETF) . Pridobljeno: 10. februar 2025. [Na spletu]. Dostopno na: https://datatracker.ietf.org/doc/draft-ietf-tls-hybrid-design/
[65] D. Benjamin, »TLS Key Share Prediction - draft-ietf-tls-key-share-prediction-01«. Pridobljeno: 10. februar 2025. [Na spletu]. Dostopno na: https://datatracker.ietf.org/doc/draft-ietf-tls-key-share-prediction/
[66] M. Barbosa *idr.*, »X-Wing«, *IACR Communications in Cryptology*, let. 1, št. 1, str. 3006–5496, apr. 2024, doi: 10.62056/A3QJ89N4E.
[67] D. Connolly, P. Schwabe, in B. Westerbaan, »X-Wing: general-purpose hybrid post-quantum KEM - draft-connolly-cfrg-xwing-kem-06«. Pridobljeno: 10. februar 2025. [Na spletu]. Dostopno na: https://datatracker.ietf.org/doc/draft-connolly-cfrg-xwing-kem/
[68] D. Connolly, »Hybrid PQ/T Key Encapsulation Mechanisms - draft-irtf-cfrg-hybrid-kems-01«. Pridobljeno: 10. februar 2025. [Na spletu]. Dostopno na: https://datatracker.ietf.org/doc/draft-irtf-cfrg-hybrid-kems/
[69] G. Alagic *idr.*, »Recommendations for Key-Encapsulation Mechanisms«, jan. 2025, doi: 10.6028/NIST.SP.800-227.IPD.
[70] M. Ounsworth, A. Wussler, in S. Kousidis, »Combiner function for hybrid key encapsulation mechanisms (Hybrid KEMs) - draft-ounsworth-cfrg-kem-combiners-05«. Pridobljeno: 10. februar 2025. [Na spletu]. Dostopno na: https://datatracker.ietf.org/doc/draft-ounsworth-cfrg-kem-combiners/
[71] V. Smyslov, »RFC 9242 - Intermediate Exchange in the Internet Key Exchange Protocol Version 2 (IKEv2)«, dec. 2024. Pridobljeno: 11. februar 2025. [Na spletu]. Dostopno na: https://datatracker.ietf.org/doc/rfc9242/
[72] C. Tjhai *idr.*, »RFC 9370 - Multiple Key Exchanges in the Internet Key Exchange Protocol Version 2 (IKEv2)«, dec. 2023. Pridobljeno: 11. februar 2025. [Na spletu]. Dostopno na: https://datatracker.ietf.org/doc/rfc9370/
[73] P. Kampanakis in G. Ravago, »Post-quantum Hybrid Key Exchange with ML-KEM in the Internet Key Exchange Protocol Version 2 (IKEv2) - draft-kampanakis-ml-kem-ikev2-09«. Pridobljeno: 11. februar 2025. [Na spletu]. Dostopno na: https://datatracker.ietf.org/doc/draft-kampanakis-ml-kem-ikev2/
[74] G. Wang, »Post-quantum Hybrid Key Exchange in the IKEv2 with FrodoKEM - draft-wang-hybrid-kem-ikev2-frodo-02«. Pridobljeno: 11. februar 2025. [Na spletu]. Dostopno na: https://datatracker.ietf.org/doc/draft-wang-hybrid-kem-ikev2-frodo/
[75] P. Membrey, »ExpressVPN: Early Adopter of ML-KEM for Quantum Encryption«. Pridobljeno: 11. februar 2025. [Na spletu]. Dostopno na: https://www.expressvpn.com/blog/ml-kem-lightway-upgrade/?srsltid=AfmBOoqeaBwEW2X41uPZqYvJa44JB4OctMt_00Cy3uGipXgMUe1YO9xh
[76] A. Hülsing, K.-C. B. Ning KPN, P. Schwabe, F. Johanna Weber, in P. R. Zimmermann, »Post-quantum WireGuard«, *Cryptology ePrint Archive*, sep. 2023, Pridobljeno: 12. februar 2025. [Na spletu]. Dostopno na: https://eprint.iacr.org/2020/379
[77] R. Housley, J. Gray, in T. Okubo, »RFC 9629 - Using Key Encapsulation Mechanism (KEM) Algorithms in the Cryptographic Message Syntax (CMS)«, avg. 2024. Pridobljeno: 11. februar 2025. [Na spletu]. Dostopno na: https://datatracker.ietf.org/doc/rfc9629/
[78] M. Ounsworth, J. Gray, M. Pala, J. Klaußner, in S. Fluhrer, »Composite ML-KEM for use in X.509 Public Key Infrastructure and CMS - draft-ietf-lamps-pq-composite-kem-05«. Pridobljeno: 11. februar 2025. [Na spletu]. Dostopno na: https://datatracker.ietf.org/doc/draft-ietf-lamps-pq-composite-kem/
[79] S. Kousidis, J. Roth, F. Strenzke, in A. Wussler, »Post-Quantum Cryptography in OpenPGP - draft-ietf-openpgp-pqc-07«. Pridobljeno: 11. februar 2025. [Na spletu]. Dostopno na: https://datatracker.ietf.org/doc/draft-ietf-openpgp-pqc/
[80] E. Kret in R. Schmidt, »The PQXDH Key Agreement Protocol«. Pridobljeno: 11. februar 2025. [Na spletu]. Dostopno na: https://signal.org/docs/specifications/pqxdh/#preliminaries
[81] D. Collins, L. Huguenin-Dumittan, N. K. Nguyen, N. Rolin, in S. Vaudenay, »K-Waay: Fast and Deniable Post-Quantum X3DH without Ring Signatures«, *Cryptology ePrint Archive*, 2024, Pridobljeno: 11. februar 2025. [Na spletu]. Dostopno na: https://eprint.iacr.org/2024/120

▪

Marko Kompara je raziskovalec in asistent na Fakulteti za elektortehniko, računalništvo in informatiko, kjer je tudi pridobil doktorat z disertacijo na temo protokolov za dogovarjanje ključev v omejenih okoljih. Od takrat je bil in je še vedno vključen v številke projekte na področju kibernetske varnosti (npr. CyberSec4Europe, RUKIV, RPUP, AKADIMOS). Njegovi raziskovalni interesi so zasebnost, kriptografija, brezžične komunikacije in varnost informacijskih sistemov.

▪

Marko Hölbl raziskuje kibernetsko varnost in zasebnost na širokem področju, od kriptografije do uporabniških vidikov informacijske varnosti in zasebnosti. Trenutno je prodekan za raziskave na FERI UM, aktiven član in generalni sekretar CEPIS LSI, član ECSO (WG6), član izvršnega odbora Slovenskega društva Informatika in član Sekcije za kibernetsko varnost pri Gospodarski zbornici Slovenije. Sodeluje in/ali vodi številne projekte iz področja kibernetske varnosti (npr. CyberSec4Europe, RUKIV, RPUP, AKADIMOS)

Poslovanje pod drobnogledom - primerjalna analiza sodobnih orodij za rudarjenje procesov

Tilen Tratnjek, Gregor Polančič
Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko, Koroška cesta 46, 2000 Maribor
tilen.tratnjek@um.si, gregor.polancic@um.si

Izvleček

Dostopne primerjave med orodji za rudarjenje procesov so pogosto zastarele ali ne vključujejo ocene iz uporabniške perspektive, kar otežuje izbiro primernega orodja. Namen prispevka je bil ponuditi primerjalno analizo štirih sodobnih orodij in tako omogočiti vpogled v njihove prednosti in slabosti. Cilj je bil ugotoviti, katera orodja so najbolj primerna za poslovne uporabnike iz vidikov razpoložljivosti, naprednosti funkcionalnosti in enostavnosti uporabe.

Analiza se je izvedla z enotnim metodološkim pristopom, ki je vključeval pregled dokumentacije, praktično testiranje z dvema javno dostopnima podatkovnima nizoma ter ocenjevanjem po vnaprej določenih kriterijih.

Ugotovljeno je bilo, da orodji Celonis in Apromore ponujata najbolj celovito podporo poslovnim uporabnikom. Obe orodji dosegata najvišjo oceno naprednosti v skoraj vseh sklopih funkcionalnosti, sta enostavni za uporabo in vključujeta umetno inteligenco. Na drugi strani ProM in PM4Py omogočata večjo prilagodljivost in preglednost analitičnih postopkov, vendar zahtevata tehnično znanje in zato nista primerna za široko uporabo brez ustreznega usposabljanja. S poudarkom na praktični uporabniški izkušnji in enostavnosti uporabe prispevek dopolnjuje obstoječe analize, ki se osredotočajo predvsem na seznam funkcionalnosti.

Ključne besede: primerjalna analiza, rudarjenje procesov, enostavnost uporabe, naprednost funkcionalnosti, Apromore, Celonis, PM4Py, ProM

Business Under the Microscope – Comparative Analysis of Modern Process Mining Tools

Abstract

Available comparisons between process mining tools are often outdated or lack evaluation from a user perspective, which makes it difficult to choose a suitable tool. The purpose of this study was to provide a comparative analysis of four modern tools and offer insight into their strengths and weaknesses. The goal was to determine which tools are most suitable for business users in terms of availability, functional sophistication and ease of use.

The analysis was conducted using a consistent methodological approach, which included a review of documentation, hands-on testing with two publicly available datasets, and evaluation based on predefined criteria.

The findings show that Celonis and Apromore offer the most comprehensive support for business users. Both tools received the highest scores for functionality in nearly all categories, are easy to use, and include artificial intelligence features. On the other hand, ProM and PM4Py provide greater flexibility and transparency in analytical procedures but require technical knowledge and are therefore not suitable for widespread use without proper training.

By emphasizing practical user experience and ease of use, this study complements existing analyses that focus mainly on lists of features.

Keywords: Comparative analysis, process mining, ease of use, functional sophistication, Apromore, Celonis, PM4Py, ProM

1 UVOD

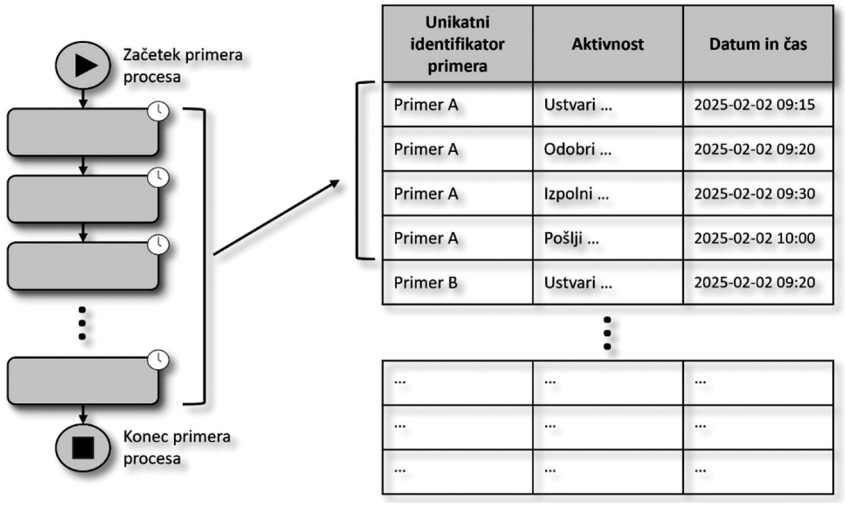
V sodobnem digitalnem okolju delujejo organizacije v kompleksnih in hitro spreminjajočih se pogojih, kjer je razumevanje in optimizacija poslovnih procesov ključnega pomena za konkurenčnost, skladnost in odzivnost. Tradicionalni pristopi k upravljanju procesov, ki temeljijo na modeliranju in subjektivnih ocenah, pogosto ne zadostujejo za celovito in objektivno analizo dejanskega poteka dela. V zadnjem desetletju se je zato uveljavilo rudarjenje procesov (angl. process mining) podatkovno podprt pristop, ki omogoča odkrivanje, preverjanje in izboljševanje procesov na podlagi dejanskih sledi dogodkov, zajetih v informacijskih sistemih.

Rudarjenje procesov predstavlja enega najhitreje rastočih segmentov poslovne programske opreme, kar potrjujejo tudi podatki raziskovalne hiše Gartner. V svojem zadnjem poročilu opisuje rudarjenje procesov kot »digitalni rentgen« celotne organizacije, ki omogoča celovit pregled poslovanja in razkriva tako odstopanja kot optimalne poteke procesov. Trg, ki je v letu 2023 že dosegel vrednost 871,6 milijona dolarjev, naj bi po napovedih do leta 2028 presegel 2 milijardi dolarjev. Poleg tega Gartner poudarja, da so ključni dejavniki tudi vse večje zahteve po operativni odpornosti (angl. operational resilience) in doseganju trajnostnih ciljev, saj optimizacija procesov neposredno zmanjšuje porabo virov in stroške [1].

S hitro rastjo področja pa se je na trgu pojavila množica specializiranih orodij, kar postavlja odločevalce pred kompleksen izziv. Izbira pravega orodja je postala ključna, a hkrati težavna. Na eni strani so odprtokodna orodja, ki tradicionalno ponujajo prilagodljivost in dostop do najnovejših akademskih algoritmov, na drugi pa komercialne platforme, ki običajno nudijo intuitivne vmesnike, celovito podporo in napredne integracije [2].

Namen tega članka je odgovoriti na naslednja raziskovalna vprašanja, ki si jih pri delu postavljajo procesni inženirji in analitiki: (1) Katere in kako napredne funkcionalnosti ponujajo izbrana orodja rudarjenje procesov? (2) Kako enostavna za uporabo so ta orodja z vidika poslovnega uporabnika? (3) Ali orodja vključujejo funkcionalnosti umetne inteligence, ki pomagajo pri interpretaciji in vrednotenju rezultatov? Na podlagi izsledkov sistematično izvedene primerjalne analize predstavljamo primerjavo štirih orodij, ki predstavljajo tako odprtokodna kot tudi komercialne rešitve. Analizo smo zasnovali z vidika povprečnega poslovnega uporabnika, ki nima programerskega znanja, a želi iz podatkov pridobiti konkretno poslovno vrednost.

Članek je v nadaljevanju strukturiran naslednje. V naslednjem poglavju so predstavljene teoretične osnove rudarjenja procesov in ključni koncepti, potrebni za razumevanje te discipline. Sledi predstavitev uporabljene metodologije ter ocenjevalnega okvira, ki omogočata sistematično primerjavo orodij. Osrednji del sestavlja analiza štirih izbranih orodij, pri čemer rezultate predstavljamo skozi dve ključni dimenziji: naprednost funkcionalnosti in enostavnost uporabe. V razpravi umestimo ugotovitve v širši kontekst, oblikujemo praktične smernice za izbiro najprimernejšega orodja ter v zaključku povzamemo ključna spoznanja raziskave.



Slika 2.1: **Ključni podatki za dnevnik dogodkov [3].**

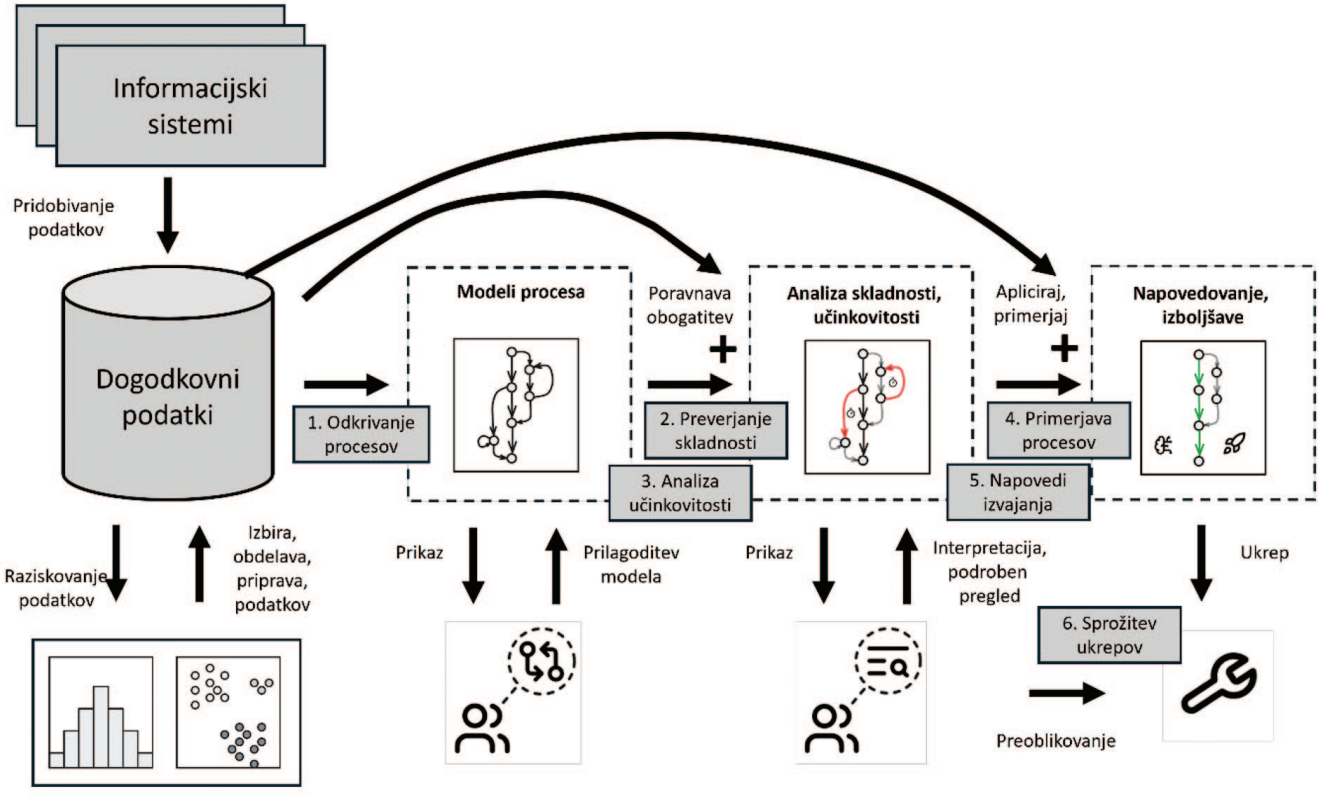
2 VPOGLED V DELOVANJE RUDARJENJA PROCESOV

Rudarjenje procesov temelji na podatkih, analitičnih tehnikah in vizualnih rezultatih, ki skupaj omogočajo objektiven vpogled v delovanje organizacije. Temelj vsake analize predstavlja dnevnik dogodkov – strukturiran zapis vseh aktivnosti v procesu. Sodobni informacijski sistemi (ERP, CRM, bolnišnični sistemi) avtomatsko beležijo vsak korak v poslovnih procesih in ustvarjajo te dnevnike dogodkov (angl. event log). Kot ponazarja Slika 2.1, mora vsak tak zapis za osnovno analizo vsebovati tri ključne informacije [2].

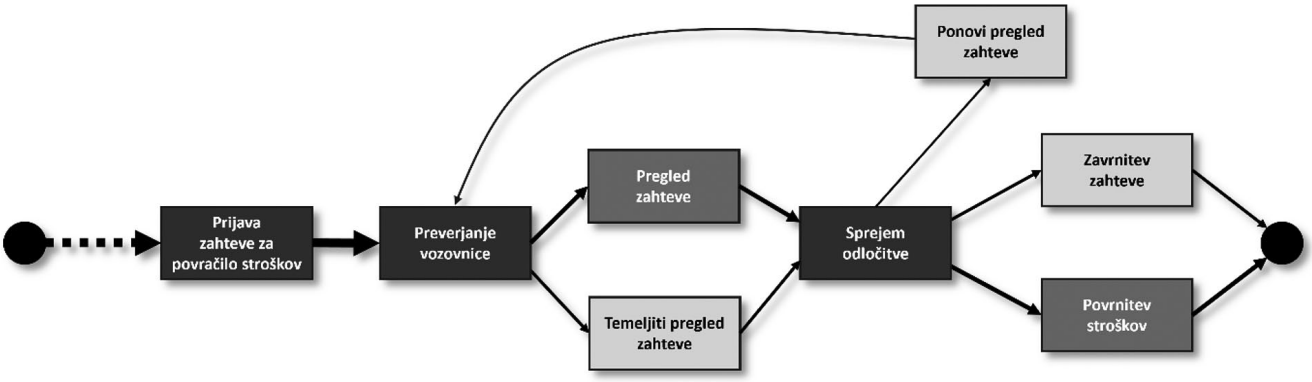
1. Unikatni identifikator primera - določa, na kateri specifičen primer se dogodek nanaša (npr. številka naročila, identifikator pacienta, številka zahtevka). Ta identifikator omogoča sledenje celotni poti posameznega primera od začetka do konca.
2. Aktivnost - opisuje konkretni korak v procesu, ki je bil izveden (npr. »prejem naročila«, »sprejem pacienta«, »pošiljanje računa«).
3. Časovni žig - beleži natančen datum in čas izvedbe aktivnosti.

Ta strukturirana zbirka podatkov predstavlja objektivno resnico o tem, kaj se v procesu dogaja, kdaj se dogaja in v kakšnem zaporedju. Dnevnik dogodkov je digitalna sled, ki omogoča znanstveno preučevanje procesov na podlagi konkretnih dejstev namesto predpostavk [3].

Ko imamo zbrane podatke, se začne jedro rudarjenja procesov, ki ga najbolje ponazarja cikel na Sliki 2.2. Ta cikel poteka v treh glavnih fazah, od katerih je prva odkrivanje procesov (angl. Process Discovery). V tej fazi algoritmi samodejno analizirajo dnevnik



Slika 2.2: **Splošni pregled rudarjenja procesov [4].**



Slika 2.3: **Notacija procesnega zemljevida na primeru procesa postopka obravnave zahtevka za povračilo stroškov v letalski družbi [2].**

dogodkov in iz njega ustvarijo vizualni model procesa. Namesto ročnega risanja procesov nam orodje na podlagi realnih podatkov samodejno ustvari vizualizacijo dejanskega poteka. S tem odgovorimo na ključno vprašanje: »Kako naši procesi dejansko potekajo?« [2].

Druga faza, preverjanje skladnosti in analiza (angl. Conformance Checking), omogoča sistematično primerjavo odkritega modela dejanskega stanja z idealnim ali predpisanim referenčnim modelom. Ta pristop razkrije ključne problematične točke: odstopanja od standardnih postopkov, preskakovanje obveznih korakov, nepotrebne ponovitve aktivnosti ter kršitve veljavnih poslovnih pravil. Vzporedno z analizo skladnosti lahko izvajamo tudi analizo učinkovitosti, s katero identificiramo ozka grla v procesu, najdemo najdaljše kritične poti ter določimo aktivnosti z največjim vplivom na celotno trajanje procesa [2].

Tretja faza predstavlja izboljšanje in napovedovanje (angl. Enhancement), kjer se spoznanja iz prejšnjih dveh faz pretvorijo v konkretne izboljšave procesov. Ker sedaj razumemo temeljne vzroke težav, lahko implementiramo ciljno usmerjene in učinkovitejše ukrepe. Ta faza omogoča tudi prehod od reaktivnega k proaktivnemu pristopu. Napredna analitična orodja namreč omogočajo napovedovanje prihodnjih dogodkov in s tem preventivno reševanje težav, še preden te sploh nastanejo [2].

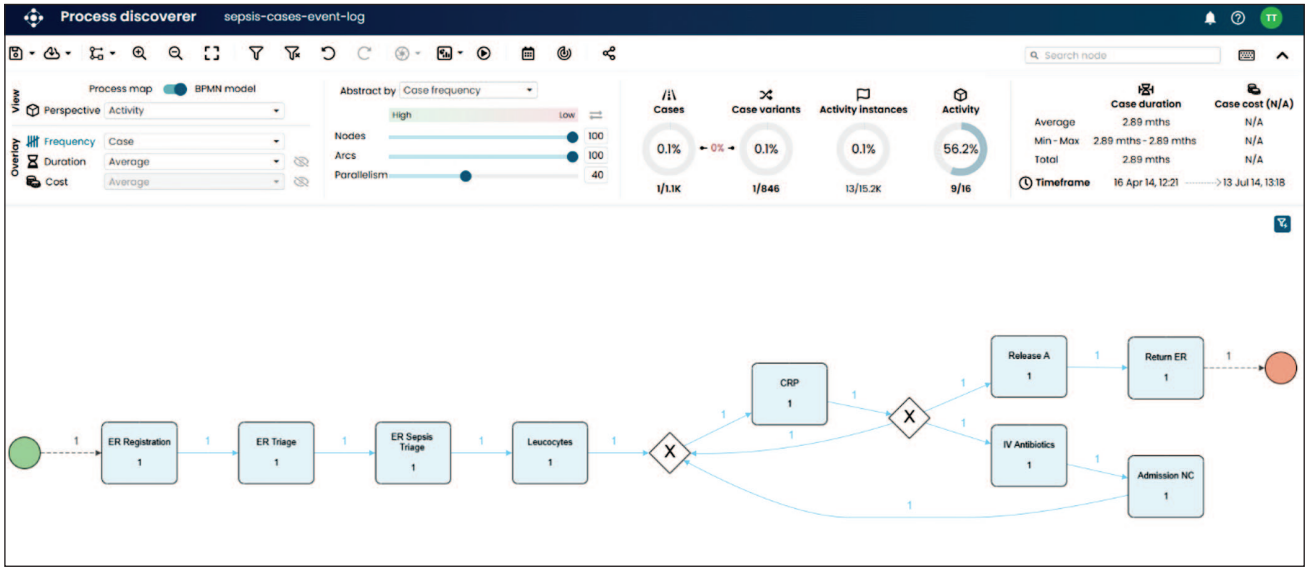
Končni rezultat odkrivanja procesov je pogosto procesni zemljevid (angl. process map), kot ga vidimo na Sliki 2.3. Ta vizualizacija na preprost in intuitiven način združi posamezne dogodke v celovito predstavo procesa, ki razkriva glavne poti izvajanja zahtevkov, alternativne tokove ter nepričakovane zanke, kot je »Ponovi pregled zahteve«. Procesni zemljevid predstavlja ključno orodje za razumevanje, saj zapleteno poslovno logiko preoblikuje v jasno vizualno predstavo, dostopno tako analitikom kot vodstvu. Kot temelj za nadaljnje analize in strateške odločitve omogoča prepoznavanje ozkih grl, neučinkovitosti in priložnosti za optimizacijo [3].

2.1 Orodja za rudarjenje procesov

Za izvedbo vseh omenjenih faz rudarjenja procesov je nujna uporaba specializiranih programskih orodij, ki obdelujejo dnevnike dogodkov, uporabljajo algoritme za odkrivanje procesov, preverjajo skladnost s predpisanimi modeli in omogočajo analizo ter izboljšave procesov. Poleg osnovnih funkcionalnosti lahko ponujajo tudi možnosti za napredno vizualizacijo, filtriranje podatkov, povezovanje z zunanjimi sistemi in uporabo metod umetne inteligence. Na trgu je na voljo širok nabor orodij [5] z različnimi pristopi in zmogljivostmi. V nadaljevanju so predstavljena štiri orodja, ki omogočajo primerjavo različnih pristopov v rudarjenju procesov, tako s tehničnega kot uporabniškega vidika. Izbor orodij na katerih je bila v nadaljevanju izvedena primerjalna analiza je temeljila na kombinaciji pregleda literature in tržnih poročil. Med komercialnimi orodji smo izbrali Celonis in Apromore, ki sodita med vodilne platforme v Gartnerjevem tržnem poročilu Magic Quadrant for Process Mining [1]. Za odprtokodni orodji smo vključili ProM in PM4Py, saj sta bile najbolj pogosto omenjeni odprtokodni orodji v pregledani literaturi.

2.1.1 Apromore Portal

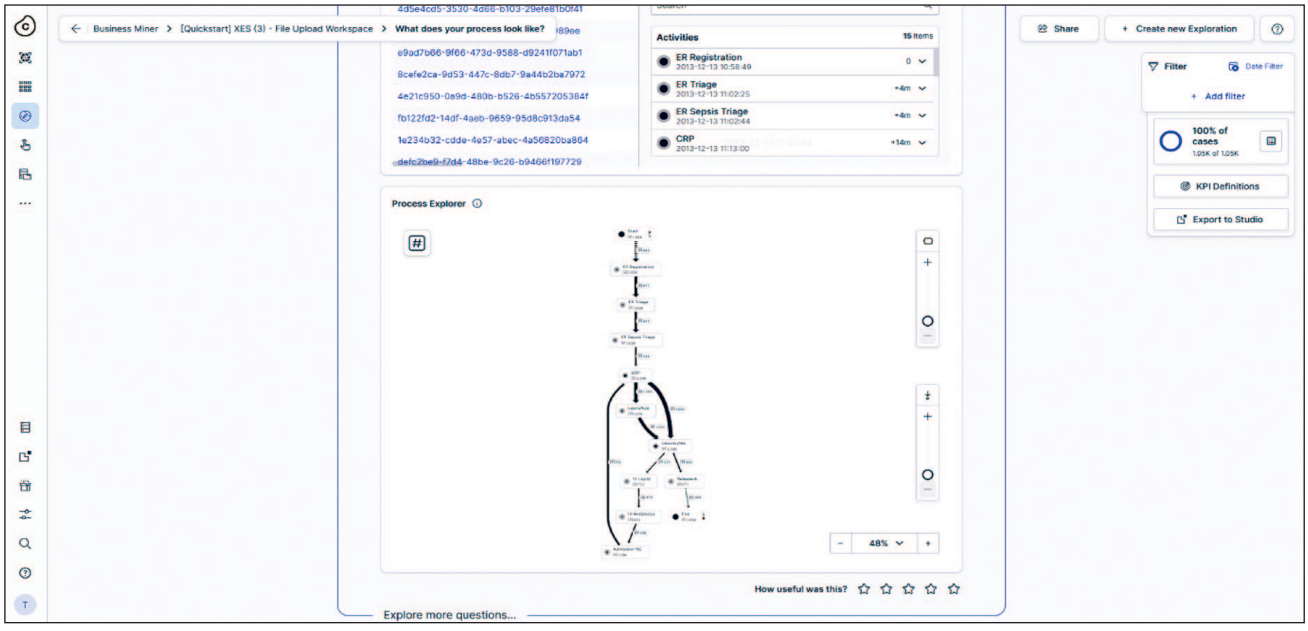
Apromore Portal 10.2 (v nadaljevanju Apromore) je komercialno licencirano orodje za rudarjenje procesov, razvito s strani podjetja Apromore Pty Ltd, ki



Slika 2.4: **Del uporabniškega vmesnika v orodju Apromore, ki prikazuje odkriti proces.**

izvira iz raziskovalnega sodelovanja med Univerzo v Melbournu in Univerzo v Tartuju. Od prve izdaje leta 2009 se je orodje neprestano razvijalo, z zadnjo posodobitvijo v marcu 2025. Uporabniki lahko dostopajo do storitve preko spletnih brskalnikov kot programsko opremo v oblaku ali z namestitvijo na lastni infrastrukturi [6].

Orodje je zasnovano za analizo in optimizacijo poslovnih procesov ter omogoča odkrivanje procesov, preverjanje skladnosti, simulacijo in napovedno analitiko. Del uporabniškega vmesnika orodja, ki prikazuje zemljevid odkritega procesa, je prikazan na Sliki 2.4. Apromore temelji na več kot desetletju inovacij in akademskih raziskav, vključno z rezultati več doktoratov in več kot 200 znanstvenih publikacij. Čeprav je bilo orodje sprva odprtokodno, zdaj deluje kot komercialno orodje s fleksibilnimi cenovnimi modeli, prilagojenimi obsegu uporabe, ter nudi tehnično podporo in redne posodobitve [4].



Slika 2.5: **Del uporabniškega vmesnika v orodju Celonis, ki prikazuje odkriti proces.**

2.1.2 Celonis Platform

Celonis Platform (v nadaljevanju Celonis) je komercialno licencirano orodje za rudarjenje procesov, ki ga razvija podjetje Celonis SE. Od prve izdaje leta 2011 se je orodje nenehno razvijalo, z zadnjo posodobitvijo v juniju 2025. Uporabniki lahko dostopajo do platforme preko spletnih brskalnikov kot programsko opremo v oblaku ali z namestitvijo na lastni infrastrukturi [7].

Celonis je namenjen analizi in optimizaciji poslovnih procesov ter omogoča izboljšanje poslovne učinkovitosti z uporabo procesne inteligence. Platforma povezuje podatke iz različnih sistemov in omogoča celosten vpogled v delovanje organizacije. Del uporabniškega vmesnika orodja, ki prikazuje zemljevid odkritega procesa, je prikazan na Sliki 2.5. Ključne zmožnosti orodja vključujejo rudarjenje procesov osredotočeno na objekte za natančen vpogled, rudarjenje nalog (angl. task mining) za razumevanje kako se opravljajo naloge, vmesnik za izgradnjo aplikacij in nadzornih plošč ter avtomatizacijo delovnih tokov za odpravljanje neučinkovitosti. Platforma omogoča tudi deljenje ustvarjenih vpogledov [7].

2.1.3 PM4Py

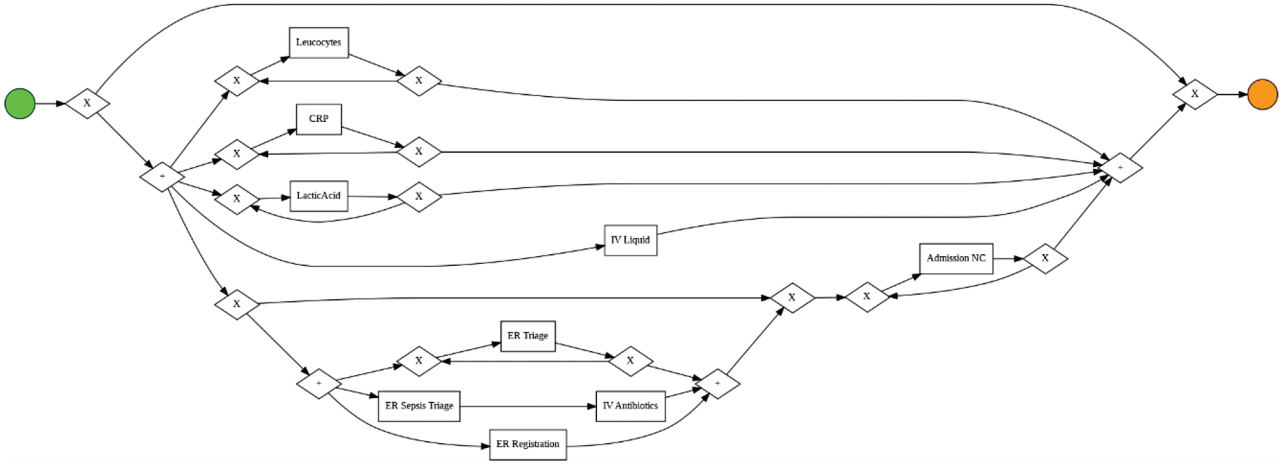
PM4Py 2. 7. 13 je odprtokodna knjižnica za rudarjenje procesov v Pythonu, ki jo razvija podjetje Process Intelligence Solutions (PIS), prvotno pa je bila razvita v okviru Fraunhofer FIT. Od prve izdaje leta 2018 se knjižnica aktivno razvija, z zadnjo posodobitvijo v juniju 2025. Kot Python knjižnica je PM4Py na voljo na vseh platformah, ki podpirajo Python programski jezik [8].

Knjižnica je namenjena tako akademskim raziskavam kot industrijski uporabi ter omogoča analizo procesnih podatkov s pomočjo različnih algoritmov rudarjenja procesov. Vključuje funkcionalnosti za odkrivanje procesov, preverjanje skladnosti in izboljševanje procesov. PM4Py podpira širok nabor algoritmov rudarjenja procesov, kot so Alpha Miner, Inductive Miner, Heuristics Miner in ILP Miner, ter omogoča generiranje različnih vrst procesnih modelov, vključno s Petrijevimi mrežami, procesnimi drevesi, procesnimi zemljevidi in BPMN diagrami. Najnovejša velika različica 2.7.0 uvaja tudi integracijo z OpenAI, kar omogoča uporabo velikih jezikovnih modelov za razlago in analizo procesnih podatkov [8]. Sli-

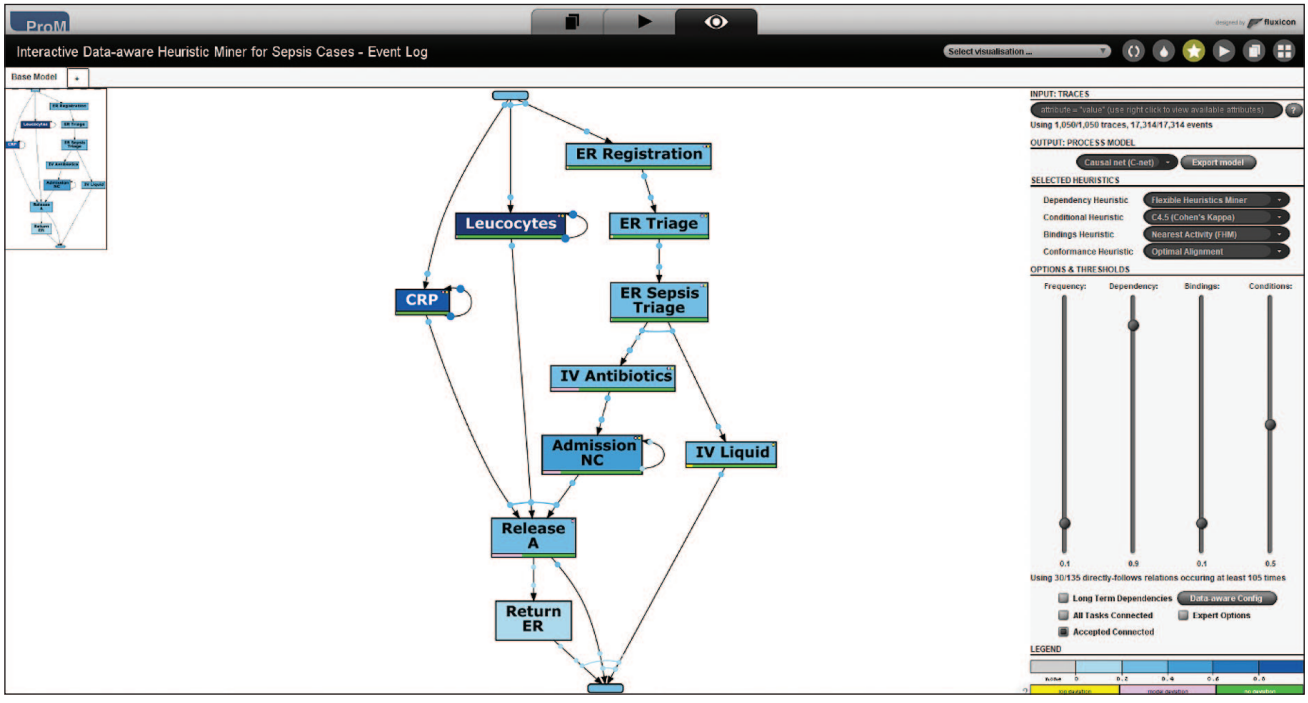
main.py

```
log = pm4py.read_xes('sepsis-cases-event-log.xes')
process_model = pm4py.discover_bpmn_inductive(log)
pm4py.view_bpmn(process_model)
```

Slika 2.6: **Prikaz odseka kode, potrebnega za odkrivanje procesa in ustvarjanje diagrama BPMN.**



Slika 2.7: **Ustvarjen diagram BPMN, ki je rezultat predstavljene kode na Sliki 2.6.**



Slika 2.8: **Del uporabniškega vmesnika v orodju ProM, ki prikazuje odkriti proces.**

ka 2.6 prikazuje odsek kode, za ustvarjanje diagrama BPMN, ki je predstavljen na Sliki 2.7.

2.1.4 ProM

ProM 6.14 je odprtokodno orodje za rudarjenje procesov, ki ga razvija Tehniška univerza v Eindhovnu (TU/e). Prvič je bilo izdano leta 2004 kot ProM 1.1, z zadnjo posodobitvijo v septembru 2023. Gre za programsko opremo, ki deluje na operacijskih sistemih Windows, Linux in macOS [2].

Orodje je namenjeno akademskemu raziskovanju in analizi procesov ter se osredotoča na odkrivanje procesov, preverjanje skladnosti in izboljševanje procesov na podlagi dnevnikov dogodkov. ProM odlikuje modularna zasnova, ki omogoča prilagajanje analitičnih metod in dodajanje novih algoritmov. Podpira različne tehnike rudarjenja procesov, med katerimi so Alpha Miner, Heuristic Miner, Fuzzy Miner in Genetic Miner. Orodje temelji na programskem jeziku Java in deluje kot lokalna aplikacija [9]. Del uporabniškega vmesnika orodja, ki prikazuje zemljevid odkritega procesa, prikazuje Slika 2.7.

3 METODOLOGIJA PRIMERJALNE ANALIZE

To poglavje predstavlja razviti okvir za ocenjevanje, ki omogoča zanesljive in ponovljive rezultate, ter podrobno opisuje uporabljeno metodologijo raziskave. Pred izvedbo primerjalne analize smo želeli najprej razumeti, kako so primerjave orodij za rudarjenje procesov zasnovali drugi avtorji. S tem smo pridobili vpogled v uporabljene metodološke pristope, kriterije primerjave in način vrednotenja funkcionalnosti posameznih orodij. Poleg tega nam je pregled sorodne literature omogočil prepoznavanje pogosto primerjanih orodij in morebitnih vrzeli v obstoječih raziskavah. To je bilo ključno za utemeljitev izbire orodij in oblikovanje primerjalnega okvira. Iskanje literature je bilo tako pomemben korak za zagotovitev metodološke utemeljenosti in relevantnosti izvedene analize.

Sorodno literaturo smo iskali sistematično z uporabo baze Google Scholar, pri čemer smo uporabili iskalni niz »intitle:(»process mining«) AND intitle:(comparison OR »comparative analysis«)« in omejitev na obdobje 2019–2025. V prvi fazi smo rezultate filtrirali glede na časovno ustreznost, vrsto raziskave ter vsebinsko ustreznost, kjer smo upoštevali le prispevke, ki obravnavajo primerjavo funkcionalnosti orodij za rudarjenje procesov. Izločili smo prispevke, ki se ukvarjajo zgolj s teoretičnimi vidiki brez obravnave konkretnih orodij.

Na tej osnovi smo izločili večino zadetkov in identificirali 18 raziskav, ki so bile potencialno relevantne. V drugi fazi smo te podrobneje pregledali in dodatno ovrednotili glede na neposredno obravnavo

primerjave orodij, aktualnost ter dostopnost celotnega besedila prek Univerze v Mariboru. Pregled literature je razkril pomembne vrzeli, kot so neaktualnost raziskav zaradi nenehnega razvoja orodij, slabše upoštevanje sodobnih trendov, kot je uporaba umetne inteligence in slaba presoja uporabniške izkušnje ob upoštevanju potreb različnih tipov uporabnikov.

Ta pristop je zagotovil, da smo obravnavali široko uporabljana in v praksi preverjena orodja, katerih pogosta omemba v literaturi potrjuje tako njihov akademski vpliv kot operativno zanesljivost v različnih organizacijskih kontekstih.

3.1 Pristop ocenjevanja

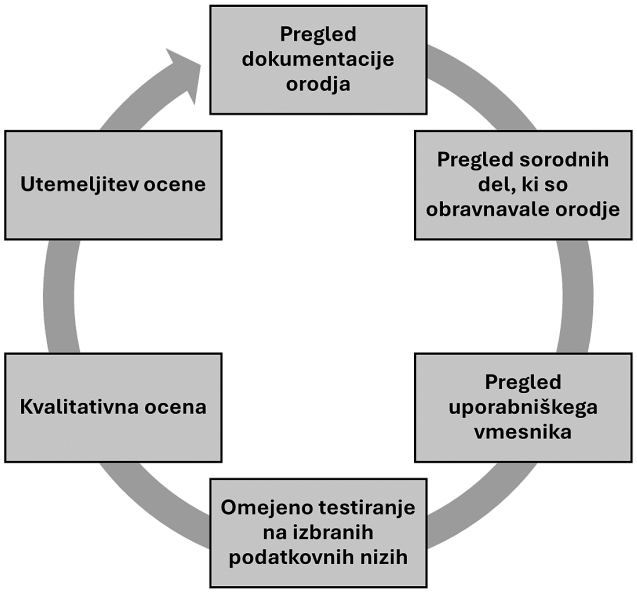
Razvili smo večstopenjski metodološki pristop, ki temelji na kombinaciji teoretičnega pregleda, praktičnega testiranja in sistematičnega ocenjevanja. Celoten proces je zasnovan kot cikel, ki omogoča iterativno poglabljanje razumevanja in utemeljevanje končnih ocen.

Vrednotenje vsakega orodja je potekalo skozi več medsebojno povezanih faz, kot je ponazorjeno na Sliki 3.1. Ta pristop zagotavlja, da končna ocena ni posledica zgolj ene aktivnosti, ampak sinteza spoznanj, pridobljenih iz različnih perspektiv.

Prvi korak je bil temeljit pregled uradne dokumentacije, uporabniških priročnikov ter obstoječih znanstvenih in strokovnih člankov, ki so obravnavali izbrana orodja. S tem smo zgradili teoretično osnovo in razumeli zmožnosti vsakega orodja. V fazi pregleda uporabniškega vmesnika smo se osredotočili na praktično izkušnjo. Analizirali smo logiko navigacije, jasnost postopkov, vizualno privlačnost in splošno intuitivnost vmesnika. Cilj je bil oceniti, kako hitro se lahko nov uporabnik znajde v orodju.

Teoretična spoznanja smo preverili v praksi z uporabo dveh, javno dostopnih podatkovnih nizov (»Primeri sepse« in »Upravljanje kazni za prometne prekrške«) [10], [11]. Izbrana sta bila, ker predstavljata procesa z različno stopnjo kompleksnosti in variabilnosti. Zdravstveni proces z visoko nepredvidljivostjo in administrativni proces z bolj strukturiranim tokom. To nam je omogočilo neposredno primerjavo, kako se orodja obnesejo pri reševanju konkretnih analitičnih nalog. V zadnji fazi smo združili vse pridobljene informacije. Na podlagi opazovanj iz pregleda dokumentacije, uporabniške izkušnje in praktičnega testiranja smo izvedli končno kvalitativno oceno po vnaprej določenih merilih. Vsaka ocena je bila argumentirano utemeljena, s čimer smo zagotovili transparentnost celotnega procesa.

Ključni del metodologije je bil enoten ocenjevalni okvir, ki je bil uporabljen za vsa štiri orodja. Vrednotenje je bilo zavestno izvedeno z vidika splošnega poslovnega uporabnika. To je posameznik z osnovnimi digitalnimi kompetencami in razumevanjem poslovnih procesov, vendar brez programerskega znanja ali ekspertnega poznavanja teorije rudarjenja procesov. V raziskavi smo sistematično preverili šest ključnih sklopov zmogljivosti bolj natančno opisanih v spodnji tabeli (Tabela 1).



Slika 3.1: **Cikel faz vrednotenja posameznega orodja.**

Tabela 1: **Sklopi funkcionalnosti in njihove ključne zmožnosti za rudarjenje procesov.**

Sklop funkcionalnosti	Ključne zmožnosti za rudarjenje procesov
Upravljanje podatkov	Omogoča celovito obdelavo izvornih podatkov, vključno s pretvorbo različnih formatov v analitično primerno obliko, čiščenje podatkov ter pripravo za nadaljnjo analizo.
Odkrivanje procesov	Sistematizira surove podatke o dogodkih v vizualne modele procesov, kar omogoča razumevanje dejanskih delovnih tokov in njihovih različic.
Preverjanje skladnosti	Zagotavlja mehanizme za primerjavo dejanskega poteka procesov z načrtovanimi ali optimalnimi scenariji, kar omogoča ugotavljanje odstopanj.
Analiza procesov	Ponuja zmožnosti za preučevanje procesov, vključno z identifikacijo neučinkovitosti, časovnih zamud in potencialov za izboljšave.
Nadzorovanje in poročanje	Omogoča kontinuirano spremljanje ključnih metrik procesov ter generiranje prilagodljivih poročil za podporo odločanju.
Podpora	Vključuje vidike uporabniške podpore, od tehnične dokumentacije do izobraževalnih programov, ki olajšajo uporabo orodja.

3.2 Ocenjevalna merila

V nadaljevanju so predstavljene tabele s podrobnimi opisi posameznih kriterijev, ki so služili kot osnova za ocenjevanje. Vsak kriterij vključuje jasno opredeljene ravni presoje, kar zmanjšuje vpliv subjektivnih ocen in zagotavlja bolj dosledno primerjavo med orodji. Tak pristop izboljša preglednost, omogoča ponovljivost vrednotenja.

3.2.1 Razpoložljivost funkcionalnosti

Prvi kriterij meri, ali je določena funkcionalnost v orodju dejansko na voljo brez dodatnih prilagoditev ali omejitev. Tabela 2 prikazuje, kako smo razdelili stopnje razpoložljivosti.

Tabela 2: **Kriterij razpoložljivost funkcionalnosti opisuje, kako dostopne in dosegljive so posamezne funkcionalnosti orodja.**

Razpoložljivost funkcionalnosti	
Ne	Funkcija ni na voljo in je ni mogoče izvesti niti s standardnimi funkcijami niti z dodatki ali obvodi.
Delno	Funkcija obstaja, vendar ima omejitve. Lahko je na voljo le v določenih različicah, zahteva dodatno plačilo, ima zmanjšane zmogljivosti, deluje prek obvodov (angl. workaround) ali zahteva programiranje po meri.
Da	Funkcija je v celoti na voljo kot del orodja. Za dostop do te funkcije niso potrebni dodatni moduli, vtičniki ali razvoj po meri.

3.2.2 Naprednost funkcionalnosti

Drugi kriterij ocenjuje, kako obsežna in prilagodljiva je posamezna funkcionalnost ter ali podpira tudi zahtevnejše primere uporabe. Stopnje naprednosti so opisane v Tabeli 3.

Tabela 3: **Kriterij naprednosti funkcionalnosti se osredotoča na kompleksnost, prilagodljivost in obseg funkcij, ki jih orodje ponuja.**

Naprednost funkcionalnosti	
Osnovno	Funkcionalnost pokriva le temeljne potrebe in minimalne zahteve. Ponuja omejeno število funkcij brez dodatnih možnosti ali prilagoditev. Zadostuje za enostavne primere uporabe, a ne ponuja prilagodljivosti za zahtevnejše scenarije.
Dobro	Funkcionalnost vključuje vse osnovne in nekaj naprednih funkcij. Ponuja zmerno raven prilagodljivosti, kar omogoča uporabo v širšem naboru scenarijev. Vključuje uporabne dodatne možnosti, ki presegajo minimalne zahteve.
Odlično	Funkcionalnost ponuja obsežen nabor naprednih funkcij in visoko stopnjo prilagodljivosti. Vključuje inovativne možnosti. Omogoča prilagajanje po meri za specifične potrebe in podpira kompleksne delovne tokove.

3.2.3 Enostavnost uporabe funkcionalnosti

Tretji kriterij se nanaša na to, kako enostavno je uporabljati funkcionalnost za uporabnika brez tehničnega znanja. Tabela 4 opredeljuje stopnje glede na zahtevnost uporabe.

Tabela 4: **Pojasnjuje, kako intuitivno je orodje za uporabnike z različnim tehničnim znanjem.**

Enostavnost uporabe funkcionalnosti	
Zahtevno	Funkcionalnost v orodju zahteva tehnično znanje ali programerske spretnosti. Vmesnik ni intuitiven, zahteva ročno konfiguracijo in obsežno usposabljanje. Dokumentacija je osnovna ali pomanjkljiva, vodeni delovni tokovi so minimalni, osnovne operacije pa zahtevajo več klikov. Kognitivna obremenitev je visoka.
Zmerno	Funkcionalnost v orodju je dostopno tudi poslovnim uporabnikom, čeprav nekaj tehničnega znanja pomaga. Vmesnik združuje intuitivne in tehnične elemente ter vključuje delno vodene delovne tokove. Dokumentacija je uporabna, osnovne operacije pa so razmeroma enostavne, a ne povsem brez trenja hitro dostopne, zahteve po učenju pa minimalne. Kognitivna obremenitev je nizka.. Kognitivna obremenitev je obvladljiva.
Enostavno	Funkcionalnost v orodju je intuitivna in uporabniku prijazna. Ne zahteva tehničnega znanja, ima dobro zasnovan vmesnik, celovite vodene delovne tokove in kontekstno pomoč. Osnovne operacije so

4 UGOTOVITVE PRIMERJAVE ORODIJ

V tem poglavju predstavljamo ključne ugotovitve primerjalne analize štirih orodij za rudarjenje procesov: Apromore, Celonis, PM4Py in ProM. Rezultati so pridobljeni na podlagi sistematičnega ocenjevanja po določenih metodoloških merilih, opisanih v prejšnjem poglavju. Ugotovitve so predstavljene v dveh ključnih dimenzijah: naprednost funkcionalnosti, in enostavnosti uporabe. Za lažjo vizualno primerjavo so rezultati prikazani na radarskih diagramih.

4.1 Naprednost funkcionalnosti orodij

Naprednost funkcionalnosti je bila ocenjena na lestvici od 1-osnovno do 3-odlično za vsak sklop funkcionalnosti. Kriterij za ocenjevanje je opisan v poglavju metodologij v Tabeli 1. Skupni rezultati, prikazani na Sliki 4.1, razkrivajo bistvene razlike med komercialnimi in odprtokodnimi orodji.

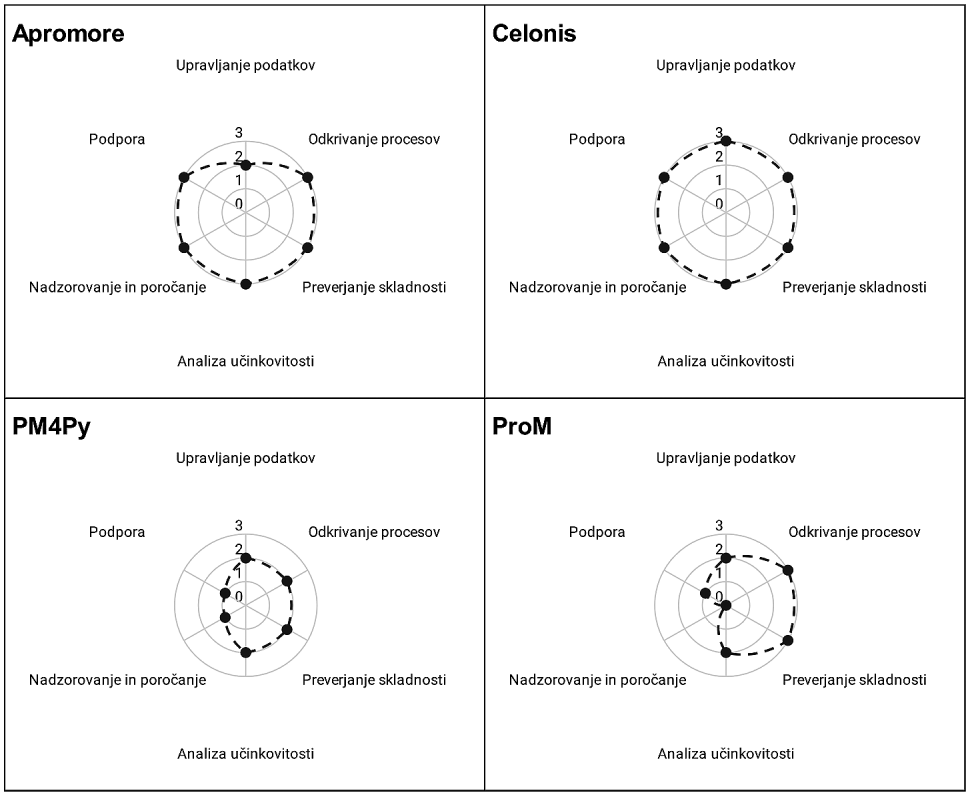
Obe komercialni platformi kažeta visoko stopnjo naprednosti v vseh ocenjevanih sklopih. Dosegata oceno odlično na področjih, kot so odkrivanje procesov, preverjanje skladnosti in nadzorovanje in poročanje. To kaže na celovito pokritost celotnega cikla rudarjenja procesov znotraj ene platforme.

Odprtokodni orodji izkazujeta bolj raznolik profil. Njihove močne točke so predvsem na področju osnovnih analitičnih zmožnosti. ProM dosega odlično oceno pri odkrivanju procesov in preverjanju skladnosti, medtem ko PM4Py dosega dobro oceno v večini analitičnih sklopov. Pri obeh odprtokodnih orodjih je opazna nižja ocena v sklopu nadzorovanje in poročanje, kjer PM4Py dosega le osnovno raven, ProM pa te funkcionalnosti v testiranem obsegu ne ponuja. Podobno nižje so ocenjene njune zmožnosti podpore.

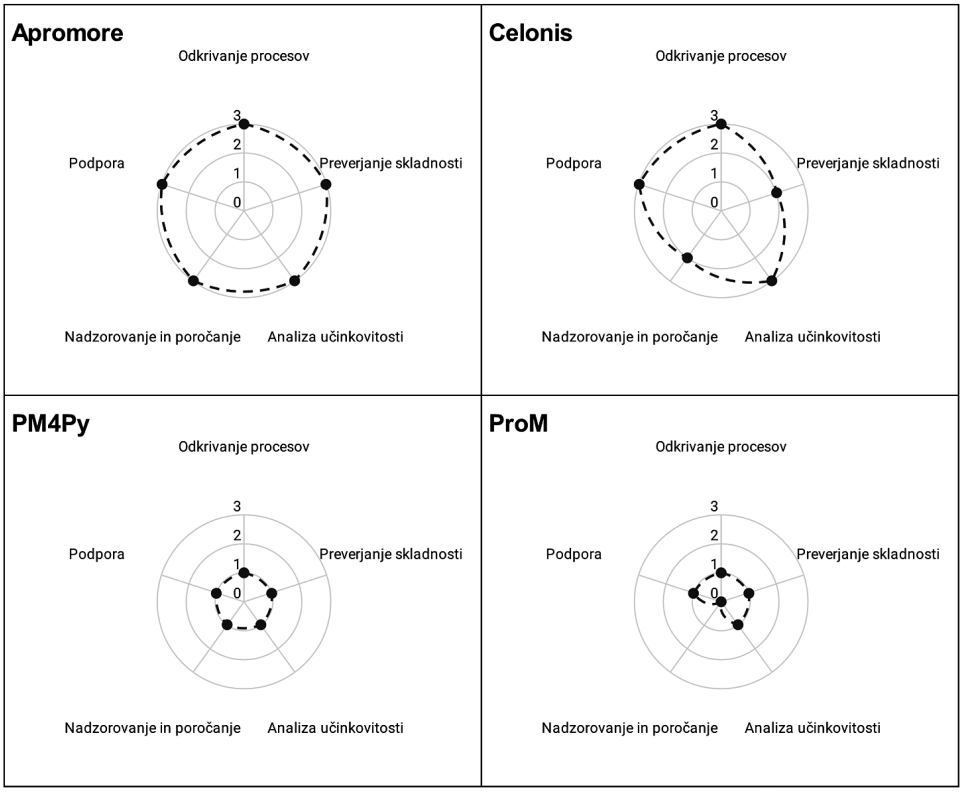
4.2 Enostavnost uporabe orodij

Enostavnost uporabe, ocenjena z vidika splošnega poslovnega uporabnika na lestvici od 1-zahtevno do 3-enostavno, razkriva še večje razlike med analiziranimi orodji. Rezultati so povzeti na Sliki 4.2.

Komercialni orodji na tem področju izrazito izstopata. Apromore dosega oceno »enostavno« v vseh kategorijah, kar kaže na visoko stopnjo intuitivnosti in nizko učno krivuljo. Celonis je prav tako ocenjen kot zelo enostaven za uporabo, vendar z nekoliko nižjo, zmerno oceno v sklopih preverjanja skladnosti ter nadzorovanja in poročanja. To je posledica



Slika 4.1 **Primerjalni diagram naprednosti funkcionalnosti izbranih orodij (1-osnovno, 2-dobro, 3-odlično).**



Slika 4.2 **Primerjalni diagram enostavnosti uporabe izbranih orodij (1-zahtevno, 2-zmerno, 3-enostavno).**

dejstva, da izgradnja naprednejših, po meri narejenih analitičnih vpogledov in nadzornih plošč poteka znotraj specifičnega okolja Celonis Studio. Gre za nizkokodno (angl. low-code) razvojno okolje, ki uporabnikom sicer omogoča izjemno prilagodljivost pri ustvarjanju lastnih vpogledov, a hkrati zahteva več tehničnega predznanja v primerjavi z osnovnimi, vnaprej pripravljenimi funkcijami platforme.

Odprtokodni orodji sta v skoraj vseh sklopih ocenjeni kot zahtevni za uporabo. ProM, z vmesnikom, ki temelji na vtičnikih, in PM4Py, ki deluje kot programska knjižnica brez grafičnega vmesnika, od uporabnika zahtevata znatno več tehničnega predznanja in samostojnega učenja za doseganje želenih rezultatov.

5 RAZPRAVA IN IMPLIKACIJE

Namen tega poglavja je razložiti ključne ugotovitve primerjalne analize, odgovoriti na raziskovalna vprašanja, opozoriti na omejitve raziskave in predlagati smeri za nadaljnje raziskovanje.

Ugotovili smo, da komercialne platforme, kot sta Apromore in Celonis, ponujajo integrirane rešitve od uvoza podatkov do vizualizacije in nadzora. Namenjene so poslovnim uporabnikom in dosegajo visoke ocene pri enostavnosti uporabe, saj se osredotočajo na intuitiven vmesnik, avtomatizacijo nalog ter dostopno predstavitev podatkov. Njihov primarni cilj je poenostavitev rudarjenja procesov in s tem večja dostopnost teh orodij tudi za netehnične uporabnike. Na drugi strani sta ProM in PM4Py, odprtokodni orodji. Njuna moč je v naprednih analitičnih zmogljivostih, visoki prilagodljivosti in možnosti razširitev. Omogočata popolni nadzor nad metodami in transparentnost postopkov, kar ju naredi primerna za akademske uporabnike, razvijalce in podatkovne znanstvenike. Vendar zahtevata več tehničnega znanja in ne dosegata enake uporabniške prijaznosti kot komercialna orodja. Te ugotovitve so skladne z obstoječim sistematičnim pregledom literature, ki prav tako poudarja razkorak med akademsko usmerjenimi in komercialnimi rešitvami. Naša analiza to potrjuje na praktičnih primerih in kvantificira razlike skozi dimenziji naprednosti in enostavnosti uporabe [12].

Rezultati so primerljivi tudi z izsledki sorodne raziskave [13], kjer so avtorji izvedli obsežno primerjalno analizo več orodij za rudarjenje procesov. Obe raziskavi izpostavljata, da ni enoznačno najboljšega

orodja, torej izbira je odvisna od konkretnega namena uporabe, tehničnega znanja uporabnikov in želenih funkcionalnosti. Ključna razlika med raziskavama je v pristopu ocenjevanja. V naši analizi je enostavnost uporabe ocenjena s praktičnim preizkusom orodij na referenčnih podatkovnih nizih in s pomočjo vnaprej določenih kriterijev. Nasprotno pa se članek [13] osredotoča samo na prisotnost funkcionalnosti, ki jih predstavlja v strukturirani tabelarni obliki, brez neposredne ocene enostavnosti uporabe.

Podobno velja za pregledan članek [14], ki analizira 16 orodij in ugotavlja, da Celonis dosega najvišjo skupno oceno glede na število podprtih funkcionalnosti. Vendar tudi ta članek ne razlikuje med pomembnostjo posameznih funkcij in vse funkcionalnosti obravnava enakovredno, ne glede na njihovo relevantnost za določene skupine uporabnikov. Poleg tega vidik uporabniške izkušnje ni vključen.

Kljub temu pa naš prispevek dopolnjuje vrzel v obstoječi literaturi, saj vključuje oceno enostavnosti uporabe na osnovi praktične interakcije z orodji, kar omogoča bolj konkreten vpogled v dejansko uporabniško izkušnjo in je še posebej pomembno za poslovne uporabnike brez tehničnega znanja.

5.1 Odgovori na raziskovalna vprašanja

1. Katere in kako napredne funkcionalnosti ponujajo izbrana orodja rudarjenje procesov?

Primerjava izbranih orodij pokaže, da komercialni orodji ponujata celovito podporo v vseh izbranih funkcionalnih sklopih rudarjenja procesov. Obe platformi dosegata najvišjo oceno naprednosti pri odkrivanju procesov, preverjanju skladnosti, analizi, nadzoru in poročanju. Razlika se pojavi pri upravljanju podatkov, kjer Celonis doseže oceno odlično, medtem ko Apromore doseže oceno dobro. Čeprav Apromore ponuja večino osnovnih in nekaj naprednih zmožnosti, integracija zunanjih podatkovnih virov ni tako razvita kot pri Celonisu. Obe orodji imata odlično urejen sistem podpore, vključno z dokumentacijo, izobraževanji in pomočjo uporabnikom. Orodji PM4Py in ProM, ki sta odprtokodni, sta močna predvsem pri analitičnih nalogah. ProM ponuja zelo napredne funkcionalnosti za odkrivanje procesov in preverjanje skladnosti, PM4Py pa dobro pokriva večino osnovnih potreb. Slabost obeh je omejena podpora za poročanje, manjša integracija podatkovnih virov in slabša uporabniška podpora. V splošnem torej komercialna orodja ponujajo bolj uravnoteženo in napredno funkcionalnost za vsakdanjo poslovno uporabo, medtem ko odprtokodna orodja omogočajo večjo prilagodljivost in raziskovalno globino, a na račun dostopnosti.

2. Kako enostavna za uporabo so ta orodja z vidika poslovnega uporabnika?

Med analiziranimi orodji izstopa Apromore, ki je bil ocenjen kot najbolj enostaven za uporabo. Njegov uporabniški vmesnik je zelo intuitiven, postopki so jasno vodeni, osnovne funkcionalnosti pa ne zahtevajo tehničnega znanja. Prav tako omogoča hitro učenje in samostojno delo brez podpore. Celonis prav tako nudi zelo visoko raven uporabnosti, vendar je za uporabo naprednejših funkcij, kot so prilagojene nadzorne plošče, potrebno osnovno razumevanje njihovega nizko kodnega (angl. low-code) razvojnega okolja Celonis Studio. Vseeno pa platforma poslovnim uporabnikom omogoča izvedbo večine analiz brez programiranja. Nasprotno pa odprtokodna orodja, kot sta ProM in PM4Py, zahtevata znatno več tehničnega znanja. ProM ima sicer grafični vmesnik, vendar je njegovo delovanje zasnovano na sistemu vtičnikov, ki pogosto ni intuitiven. PM4Py pa deluje kot Python knjižnica brez uporabniškega vmesnika, kar pomeni, da je dostopen le uporabnikom z znanjem programiranja. Za poslovne uporabnike brez tehnične podlage sta torej ta dva orodja manj primerna.

3. Ali vključujejo funkcionalnosti umetne inteligence, ki pomagajo pri interpretaciji in vrednotenju rezultatov?

Uporaba umetne inteligence v obravnavanih orodjih je še vedno različno razvita. Celonis trenutno ponuja najnaprednejšo integracijo AI funkcij. Vključuje mehanizme strojnega učenja za napovedovanje in priporočila, avtomatsko odkrivanje anomalij ter generativno AI za pomoč pri interpretaciji rezultatov. Te funkcionalnosti so dostopne tudi poslovnim uporabnikom in delujejo kot orodje za podporo odločanju. Apromore vključuje funkcije umetne inteligence vendar obseg uporabe umetne inteligence ni tako širok kot pri Celonisu. Orodji PM4Py in ProM vključujeta algoritme, ki temeljijo na konceptih strojnega učenja, vendar so ti praviloma dostopni le preko programskih vmesnikov ali vtičnikov. Nimata integriranih AI-funkcionalnosti, ki bi bile dostopne netehničnim uporabnikom. Uporaba umetne inteligence v odprtokodnih orodjih je torej mogoča, vendar le za uporabnike z znanjem programiranja.

5.2 Katero orodje izbrati?

Na podlagi rezultatov analize lahko organizacijam ponudimo naslednje smernice. Komercialna orodja (Celonis, Apromore) so primerna za organizacije, ki želijo hitro implementacijo, enostavno uporabo in široko vključevanje poslovnih uporabnikov. Uporabniški vmesnik je intuitiven, postopki so vodeni, vizualizacije dostopne tudi brez tehničnega znanja. Tovrstna orodja so posebej učinkovita v okoljih z omejenimi tehničnimi viri ali tam, kjer analitiko izvajajo predvsem poslovni uporabniki. Celonis izstopa tudi po naprednih AI-funkcionalnostih in omogoča stalno sinhronizacijo s podatkovnimi viri, kar je ključno za redno spremljanje procesov.

Odprtokodna orodja (ProM, PM4Py) omogočajo večjo prilagodljivost in metodološko globino. Primerna so za okolja z višjo stopnjo tehnične usposobljenosti, kot so raziskovalni oddelki, akademske ustanove ali podatkovne ekipe. Uporabnikom omogočajo natančen nadzor nad postopki, razvoj novih algoritmov ter integracijo analitike v lastne rešitve. Vendar pa zahtevajo več priprav, konfiguracije in programiranja, zato so bolj primerna za občasne, projektno usmerjene analize.

Hibridni pristop se kaže kot najbolj učinkovit. Organizacija lahko uporabi PM4Py za napredno pripravo in čiščenje podatkov, nato pa dnevnik naloži v Apromore za dostopno vizualizacijo in vsakodnevno spremljanje s strani poslovnih uporabnikov. Tak pristop združuje prednosti obeh svetov: tehnično globino in širšo uporabnost.

Frekvenca uporabe je še en pomemben kriterij. Komercialna orodja so zasnovana za redno, tudi vsakodnevno uporabo. Podpirajo sprotno osveževanje podatkov, nadzorne plošče v realnem času in vnaprej pripravljene metrike. Nasprotno pa odprtokodna orodja zahtevajo več priprav in so zato bolj primerna za periodične analize z jasnimi cilji, kot je iskanje ozkih grl ali testiranje hipotez.

Dostopnost in širša uvedba sta tesno povezani z uporabniško izkušnjo. Višja ocenjena enostavnost uporabe pri Celonisu in Apromoru pomeni krajši čas uvajanja, manjšo potrebo po IT podpori in večjo samostojnost uporabnikov. Ta lastnost omogoča širšo uporabo v organizaciji. Orodji ProM in PM4Py pa sta zaradi višje zahtevnosti primerna le za bolj tehnično usposobljene skupine, kar lahko upočasni širšo uvedbo.

5.3 Omejitve in prihodnje raziskave

Analiza je bila omejena na štiri izbrana orodja in dva javno dostopna podatkovna niza. Zaradi omejenega nabora scenarijev rezultati niso nujno posplošljivi na vsa orodja ali vse vrste poslovnih procesov. Nekatere funkcionalnosti bi lahko bolje prišle do izraza pri drugih vrstah podatkov ali v specifičnih industrijah.

Ena ključnih metodoloških omejitev raziskave je subjektivnost ocenjevanja. Vrednotenje enostavnosti uporabe in naprednosti funkcionalnosti je izvedel en ekspertni ocenjevalec na podlagi predhodno določenih kriterijev in enotnega metodološkega pristopa. Ta pristop sicer zagotavlja notranjo konsistentnost ocen, saj isti posameznik ohranja enak interpretacijski okvir skozi celoten postopek, a hkrati omejuje prenosljivost rezultatov. Ocenjevanje je temeljilo na osebni presoji, ki jo lahko oblikujejo posameznikove preference, izkušnje in razumevanje posameznih pojmov.

Posebej to velja za kriterij naprednosti funkcionalnosti. Uporabljene so bile tri vnaprej opredeljene ravni (osnovno, dobro, odlično), vendar brez točno določenega seznama pričakovanih funkcij za posamezno stopnjo. Zato je moral ocenjevalec presoditi, ali nabor funkcij zadostuje za zahtevnejše primere uporabe, kar dodatno povečuje tveganje za pristranskost.

Za večjo zanesljivost rezultatov bi bilo v prihodnje smiselno vključiti več ocenjevalcev z različnim tehničnim ozadjem, na primer poslovne uporabnike, analitike in raziskovalce. Poleg tega bi bilo priporočljivo izvesti strukturirano uporabniško testiranje na enotnih nalogah ter uporabiti kvantitativne metrike uporabnosti, kot so čas za izvedbo naloge ali število klikov. K večji objektivnosti bi prispevala tudi priprava natančnejših funkcionalnih kriterijev za vsako ocenjevalno kategorijo ter primerjalna validacija z dokumentacijo orodij ali izkušnjami končnih uporabnikov. Kljub navedenim omejitvam analiza ponuja uporabne smernice, temelji na enotni ocenjevalni shemi in omogoča primerljivost znotraj izbranega nabora orodij.

V prihodnjih raziskavah bi bilo smiselno razširiti nabor primerjanih orodij in vključiti bolj raznolike podatkovne nize iz različnih industrij. Pomembno bi bilo tudi preučiti podporo za napredne pristope, kot je rudarjenje procesov osredotočeno na objekte (angl. Object-Centric Process Mining, OCPM), ter poglobiti analizo vloge umetne inteligence pri interpretaciji, avtomatizaciji in širši dostopnosti procesne analitike.

5.4 Zaključek

Primerjalna analiza štirih orodij za rudarjenje procesov je pokazala, da univerzalno najboljše orodje ne obstaja. Ključni dejavniki pri izbiri so tehnično znanje uporabnikov, namen uporabe in razpoložljivi viri. Rezultati kažejo jasno ločnico med komercialnimi in odprtokodnimi orodji. Celonis in Apromore sta pokazala visoko stopnjo uporabnosti in funkcionalne pokritosti, kar ju naredi primerna za vsakodnevno uporabo v poslovnih okoljih. ProM in PM4Py sta bila ocenjena kot primernejša za raziskovalno rabo in napredne analize, a zahtevata več tehničnega znanja in nista dostopna širšemu krogu uporabnikov.

Odgovori na raziskovalna vprašanja kažejo, da komercialna orodja, kot sta Celonis in Apromore, pokrivajo vse ključne funkcionalne sklope in ponujajo uporabniku prijazne vmesnike, kar omogoča enostavno uporabo tudi brez tehničnega predznanja. Odprtokodna orodja, kot sta ProM in PM4Py, omogočajo večjo raziskovalno globino in prilagodljivost, a so zaradi večje tehnične zahtevnosti manj dostopna širšemu krogu uporabnikov. Obe komercialni orodji vključujeta tudi funkcionalnosti umetne inteligence, pri čemer Celonis trenutno ponuja širši nabor funkcij.

Iz vidika enostavnosti uporabe so komercialna orodja primerna izbira za podjetja, ki želijo hitro uvesti rudarjenje procesov z minimalnim vložkom v usposabljanje. Po drugi strani bodo organizacije z razvitimi podatkovnimi ekipami in specifičnimi raziskovalnimi cilji našle večjo fleksibilnost v odprtokodnih orodjih. Hibridna raba pa je smiselna tam, kjer je potrebno združiti analitično natančnost z dostopno vizualizacijo.

Ta primerjalna analiza ponuja konkreten vpogled v izbiro orodij za rudarjenje procesov glede na potrebe organizacij. Poudarek na uporabniški izkušnji in tehnični zahtevnosti dopolnjuje obstoječe analize, ki se večinoma osredotočajo le na prisotnost funkcij. Rezultati lahko služijo kot smernica podjetjem, ki uvajajo rudarjenje procesov ali iščejo primernejše orodje.

LITERATURA

[1] T. Srivastava, M. Kerremans, in D. Sugden, »Magic Quadrant for Process Mining Platforms«, Gartner, Inc., G00816659, apr. 2025.

[2] W. Van Der Aalst, *Process Mining*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016. doi: 10.1007/978-3-662-49851-4.

[3] S. Kaelble, *Process Mining For Dummies®, Celonis Special Edition*. John Wiley & Sons, Inc., 2022.

[4] L. Reinkemeyer, »Status and Future of Process Mining: From Process Discovery to Process Execution«, v *Process Mining Handbook*, W. Van Der Aalst in J. Carmona, Ur., Cham: Springer International Publishing, 2022, str. 405–415. doi: 10.1007/978-3-031-08848-3_13.

[5] Process and Data Science Group - RWTH Aachen University, »Software - Process Mining«, Software. Pridobljeno: 28. julij 2025. [Na spletu]. Dostopno na: https://www.processmining.org/software.html

[6] Apromore, »Business Process Mining Apromore«, Landing page. Pridobljeno: 20. februar 2025. [Na spletu]. Dostopno na: https://apromore.com/

[7] Celonis SE, »Process Intelligence and Process Mining«, Landing page. Pridobljeno: 20. februar 2025. [Na spletu]. Dostopno na: https://www.celonis.com/

[8] Process Intelligence Solutions, »PM4Py's Documentation!«, Documentation. Pridobljeno: 24. januar 2025. [Na spletu]. Dostopno na: https://processintelligence.solutions/static/api/2.7.11/index.html

[9] H. M. W. Verbeek, »ProM 6: The Process Mining Toolkit«.

[10] M. de Leoni in F. Mannhardt, »Road Traffic Fine Management Process«. Eindhoven University of Technology, 2015. doi: 10.4121/uuid:270fd440-1057-4fb9-89a9-b699b47990f5.

[11] F. Mannhardt, »Sepsis Cases - Event Log«. Eindhoven University of Technology, 2016. doi: 10.4121/uuid:915d2bfb-7e84-49ad-a286-dc35f063a460.

[12] C. A. Kesici, N. Ozkan, S. Taşkesenlioglu, in T. G. Erdogan, »A Systematic Literature Review of Studies Comparing Process Mining Tools«, *IJITCS*, let. 14, št. 5, str. 1–14, okt. 2022, doi: 10.5815/ijitcs.2022.05.01.

[13] T. BABÄLOVÄ, »Process Mining Software Evaluation«, Masaryk University, Faculty of Informatics.

[14] O. Loyola-González, »Process mining: software comparison, trends, and challenges«, *Int J Data Sci Anal*, let. 15, št. 4, str. 407–420, maj 2023, doi: 10.1007/s41060-022-00379-0.

Tilen Tratnjek je magister informatike in podatkovnih tehnologij. Njegovi raziskovalni interesi so povezani s presekom procesne in podatkovne znanosti ter učinkovite rabe umetne inteligence.

Gregor Polančič znanstveni svetnik in izredni profesor na področju informatike. Spada med vodilne raziskovalce na področjih modeliranja poslovnih procesov in konceptualnega modeliranja v informatiki, tehnik in tehnologij upravljanja poslovnih procesov, tehnologij komuniciranja in sodelovanja. Bil je gostujoči profesor in raziskovalec na številnih tujih akademskih institucijah, recenzent številnih vrhunskih znanstvenih revij in član več odborov znanstvenih srečanj. Njegova bibliografija obsega več kot 300 zapisov, od tega preko 30 izvirnih znanstvenih člankov s faktorjem vpliva.

Standardi in skladnost v IT projektih: integracija standardov ISO 27001/22301/9001 v vodenje projektov

Ester Bradač
bradac.ester@gmail.com

Izvleček

Članek obravnava integracijo standardov ISO 27001, ISO 22301 in ISO 9001 v vodenje IT projektov. Poudarja, da hkratna uporaba teh standardov prispeva k večji učinkovitosti, skladnosti, varnosti in odpornosti projektov. Na podlagi pregleda literature in izzivov iz prakse je predstavljen model, ki te standarde umešča v vse faze projektnega cikla. Integracija omogoča boljše upravljanje tveganj, višjo kakovost izvedbe ter večje zaupanje deležnikov, zato se predlaga sistematičen pristop z integracijo kot ključno razvojno usmeritev za projektno vodenje v kompleksnih IT okoljih.

Ključne besede: informacijska varnost, ISO standardi, IT projekti, kakovost, odpornost, projektno vodenje.

Standards and Compliance in IT Projects: Integrating ISO 27001/22301/9001 into Project Management

Abstract

The article discusses the integration of ISO 27001, ISO 22301, and ISO 9001 standards into IT project management. It highlights that the simultaneous application of these standards enhances project efficiency, compliance, security, and resilience. Based on a literature review and practical challenges, a model is proposed that incorporates the standards into all phases of the project lifecycle. This integration supports better risk management, higher execution quality, and increased stakeholder trust, making a systematic approach to standard integration a key development direction for managing complex IT projects.

Keywords: Information security, ISO standards, IT projects, quality, resilience, project management.

1 UVOD

Vodenje IT projektov danes presega zgolj tehnično izvedbo rešitev – postaja ključni dejavnik zagotavljanja skladnosti z regulativo, upravljanja tveganj in zagotavljanja varnosti informacij. IT okolja v zasebnem sektorju pogosto delujejo v dinamičnih in visoko reguliranih kontekstih, kjer lahko neusklajenost z mednarodnimi standardi pomeni visoko poslovno tveganje. Zaradi narave projektnega vodenja, ki ga [1] opredeljujemo kot časovno omejen in ciljno usmerjen proces, katerega namen je ustvariti edinstven izdelek, storitev ali rezultat v visoko dinamičnem in tveganem okolju, je smiselno sistematično vključevati ISO standarde v posamezne faze projektnega cikla. V Sloveniji so med najbolj razširjenimi prav standardi ISO 9001, ISO 27001 in ISO 22301, ki po pojavnosti sodijo med prvih pet [2].

Članek se odziva na potrebo po sistematični integraciji predvsem ISO 27001, ISO 22301 in ISO 9001 organizacijskih standardov v projektni cikel, saj ima

njihova implementacija neposreden vpliv na samo projektno delo. Predlagani model pa temelji na realnih izzivih iz prakse ter dopolnjuje obstoječe teorije s konkretnimi priporočili za projektne vodje.

2 TEORETIČNA IZHODIŠČA

2.1 Vključeni standardi

Organizacijam že zadnjih petdeset let ne uspeva uspešno zaključevati projektov, nenehno si prizadevajo izboljšati metode vodenja projektov, da bi dosegle večjo stopnjo uspešnosti [3]. Medtem pa postaja poslovno okolje vedno bolj podatkovno usmerjeno [4]. Za poslovno okolje kot tako, je pomemben dejavnik varovanje in upravljanje s podatki, kar lahko zagotavljamo skozi informacijsko varnost ISO 27001 standarda [5], ki se po podatkih ISO [2] uvršča na četrto mesto največkrat implementiranega standarda v svetu. Organizacijam zagotavlja smernice za vzpostavitev sistema za upravljanje s tveganji, povezanimi predvsem z varnostjo podatkov [6]. V vodenje projektov je v kontekstu neprekinjenega poslovanja smiselno vključevati tudi ISO 22301 standard, ki podjetjem omogoča sistem za zagotavljanje neprekinjenega poslovanja in se predvsem osredotoča na vpliv motenj in obnovitvenih strategij [7]. To nam omogoča prepoznavanje najbolj kritičnih funkcij organizacije ter posledic, ki jih lahko morebitni nepričakovan dogodek prinese, obenem pa omogoča določitev ukrepov za omilitev vpliva takšnega nepričakovanega dogodka – tudi v primerih, ko so ljudje, sistemi, objekti, dobavitelji in partnerji lahko dalj časa nedosegljivi [8]. Prav tako pomemben standard, ki ga je potrebno vključiti v projektno vodenje je ISO 9001 [9], standard kakovosti vodenja, s poudarkom na kakovostne procese, zadovoljstvo strank, obvladovanje sprememb in izboljšave. To je standard, ki ima po raziskavi, opravila sta jo Safder in Yousaf [10], pomemben vpliv na projektno delo. Opravljena raziskava je potrdila, da je projektno vodenje bistveno boljše v organizacijah s certifikatom ISO 9001, kot v tistih ki tega certifikata nimajo. Vseeno pa je potrebno izpostaviti dejstvo, da so standardi pisani na organizacijskem nivoju in ne na projektnem. Kljub temu pa ima certifikacija v organizaciji neposreden vpliv na uspešnost in vodenje projektov ter, da obstaja močna povezava med projektnim vodenjem in uspešnostjo organizacije in okvirom za vodenje projektov (PMBoK, Prince2) [11].

2.2 Strokovni članki

Mednarodni standard ISO 21502 [12] definira projekt kot začasno organizacijo, ustanovljeno z namenom ustvarjanja enega ali več rezultatov. Standard tudi poudarja, da projekti prispevajo k doseganju strateških ciljev organizacije in da vodenje teh projektov vključuje vodstvene naloge, odločanje, vodenje virov in nadzor napredka za dosego načrtovanih ciljev. Project managmenet institut pa v svojem PMBoK guide (7th edition) [1] projektno vodenje definira kot

Tabela 1: **Metodologije in metodološki okviri**

Metodologija/ metodološki okvir	Opis metodološkega okvirja	Področja uporabe
Agile	Metodološki okvir za prilagodljivo in iterativno delo. Osredotoča se predvsem na sodelovanje in dostavo vrednosti ter prilagajanje spremembam [14].	Informacijske tehnologije, razvoj programske opreme, okolja z visokim tempom sprememb.
Scrum	Agilna metoda s strukturiranim pristopom z opredeljenimi vlogami in časovno omejenimi iteracijami. Namenjenag povečanju preglednosti, hitrejšemu odzivu na spremembe in izboljševanju procesov. [15].	Razvoj programske opreme, agilne razvojne ekipe, organizacije z visoko stopnjo samoorganizacije.
Lean	Osredotoča se na odpravo nepotrebnega dela, izboljšanje vrednosti za končnega uporabnika in uveljavlja načelo neprekinjenega izboljševanja [16].	Proizvodnja, storitvene dejavnosti, upravljanje informacijskih sistemov.
Kanban	Vizualno orodje za upravljanje dela v realnem času. Poudarek na omejevanju obsega dela v teku (WIP) in pretočnosti [17].	Operativni in vzdrževalni procesi, razvoj programske opreme, IT podpora.
Six Sigma	Metodologija za izboljševanje kakovosti s poudarkom na zmanjševanju napak [18].	Proizvodna industrija, upravljanje kakovosti, optimizacija poslovnih procesov.
Waterfall	Tradicionalni zaporedni pristop s fiksnimi fazami (analiza, načrtovanje, izvedba, testiranje, zaključek) [19].	Inženirski in gradbeni projekti, informacijski sistemi s stabilnimi zahtevami.

uporabo znanja, veščin, orodij in tehnik za izvajanje projektnih aktivnosti z namenom izpopolnjevanja projektnih zahtev. Projektno vodenje se nanaša na usmerjanje projektnega dela za dosego želenih rezultatov. Projektne ekipe pa lahko zastavljene cilje dosežejo z uporabo različnih pristopov (npr. prediktivni, hibridni in prilagodljivi).

Metodološki okvir PRINCE2 [13] projektno vodenje definira kot načrtovanje, delegiranje, spremljanje in nadzor vseh vidikov projekta, ter motiviranje vključenih, da dosežejo projektni cilj znotraj pričakovanih mej za čas, stroške, kakovost, obseg, koristi in tveganja.

V tabeli 1 so z namenom boljše primerjave njihovih značilnosti in uporabe v praksi, zbrane še nekatere druge ključne metodologije in metodološki okviri, ki se uporabljajo pri vodenju projektov.

2.3 Pregled literature

V zadnjih dveh desetletjih se je v akademskem in strokovnem prostoru močno razširila uporaba mednarodnih standardov za izboljšanje kakovosti, varnosti in odpornosti v organizacijah, zlasti v okviru IT projektov [4], [20]. Raziskave so se večinoma osredotočale na posamezne standarde, pri čemer so analizirale njihove koristi, izzive implementacije in vpliv na uspešnost projektov.

Večina obstoječih študij obravnava učinek posameznega ISO standarda. Kitsios idr. [21] so naprimer analizirali uporabo standarda za informacijsko varnost ISO27001 v mednarodnem IT podjetju, ter prišli do zaključka, da ima standard pomembno vlogo pri varnem ravnanju s podatki v okoljih z visoko stopnjo projektne dinamike, predvsem pa se je stopnja zaupanja strank po vpeljavi tega standarda v organizaciji dvignila. Prav tako se večina obstoječih študij osredotoča na implementacijo posameznega standarda in preučuje njegove posamezne koristi, kot so izboljšana skladnost, zmanjšanje tveganj ali povečanje zadovoljstva strank [22], [23].

Ingason [24] je raziskoval področje vpeljave ISO 9001 standarda v organizacijah. S pregledom 21 organizacij po vpeljavi ISO 9001 standarda je ugotovil, da so pri sami implementaciji ključno vlogo imeli projektne vodje in uporabljeni projektni pristop. Ključni dejavniki uspeha implementacije so bili podpora, neposredna vključenost managementa in aktivna udeležba zaposlenih. Zaključek avtorjev je, da dobra priprava in organizacija - uporaba pravega projektnega pristopa – pomembno prispevata k uspešnosti, saj so organizacije z uporabo projektnega vodenja postopke zaključile v povprečju 13 mesecev, med tem ko so organizacije, ki tega pristopa niso uporabljale, ta postopek v povprečju zaključile v 24 mesecih. Med tem pa Salida in Taibi [20] v pregledu literature ugotavljata, da ne moremo govoriti o enotnem vplivu certifikacije na uspešnost organizacij. Kljub temu, da izvedena študija kaže na pozitivne učinke, menita, da so za vzpostavitev te povezave potrebni določeni pogoji.

Strelicz in Bognar [25] v sklopu ISO 22301 standarda dajeta poudarek na pripravi modela oziroma metode za ocenjevanje vpliva incidentov na poslovanje (Business Impact Analysis – BIA), ki je ključen za projektno fazo načrtovanja, predvsem z vidika odpornosti. ISO 22301 zahteva, da organizacije najprej opravijo analizo in oceno vpliva motenj na poslovanje, saj je to logična in nujna izhodiščna točka. Šele, ko podjetje jasno prepozna in razume možne posledice tveganj, lahko učinkovito načrtuje ukrepe za preprečevanje, odzivanje in obnovo.

Čeprav ti prispevki pomembno osvetljujejo posamezne dimenzije, pa se večinoma ne ukvarjajo s sočasno uporabo več standardov, kot se to pogosto dogaja v realnih okoljih, zlasti v večjih IT organizacijah.

2.4 Integracija več standardov v projektih

Kljub temu da organizacije v praksi pogosto hkrati uvajajo več ISO standardov, da bi izpolnile regulativne, varnostne in kakovostne zahteve, obstaja pomembna vrzel v literaturi. Sistematična integracija standardov – denimo ISO 27001 (varnost), ISO 22301 (odpornost) in ISO 9001 (kakovost) – še ni bila dovolj raziskana v okviru projektnega vodenja.

Redki avtorji, kot naprimer Ispas idr. [26], so pokazali, da lahko integrirani sistemi vodenja pomembno izboljšajo operativno učinkovitost, usklajenost med oddelki in odpornost organizacije na motnje. Njihova raziskava potrjuje, da so takšne integracije še posebej relevantne v projektno usmerjenih okoljih, kot so IT podjetja, kjer se standardi lahko medsebojno dopolnjujejo in krepijo.

Tudi Ispas in Mironeasa [27] v svojem prispevku ugotavljata, da integracija standardov kakovosti (ISO 9001), informacijske varnosti (ISO 27001) in nepreki-njenega poslovanja (ISO 22301) presega delovanje kot ločenih standardov – namesto tega naj se vzpostavi integriran sistem vodenja, ki povezuje ključne elemente vseh treh področij. Avtorja poudarjata, da

tak celovit pristop omogoča organizacijam učinkovitejše upravljanje tveganj, skladnost z regulativo in hitrejše doseganje strateških ciljev, saj združuje različna orodja in prakse standardov znotraj projektnega vodenja.

Povzetki literature kažejo, da posamezni standardi prispevajo k določenim vidikom uspešnosti projektov (varnost, kakovost, odpornost), vendar ni dovolj raziskano, kako njihova integracija vpliva na operativno projektno vodenje, vključno z načrtovanjem, nadzorom in poročanjem. Ugotavljamo, da obstaja potreba po razvoju celostnih pristopov, ki standarde ne bodo obravnavali kot ločene okvirje, temveč kot medsebojno povezane elemente integriranega projektnega vodenja.

Ta raziskovalna vrzel odpira prostor za razvoj novih modelov upravljanja projektov, ki bodo upoštevali kompleksnost sodobnih IT okolij in potrebo po skladnosti s številnimi regulativami in poslovnimi zahtevami hkrati.

3 RAZISKOVALNI MODEL

Na podlagi ugotovitev iz pregleda literature je mogoče oblikovati raziskovalni model, ki analizira vpliv sočasne implementacije več ISO standardov (ISO 27001, ISO 22301 in ISO 9001) na operativno vodenje IT projektov. Model temelji na predpostavki, da integracija teh standardov vpliva na ključne dimenzije projektnega vodenja [1], kot so začetek, načrtovanje, izvedba, nadzor in upravljanje tveganj ter zaključek.

V tabeli spodaj je predstavljen model integracije ISO standardov v projektno vodenje. Model prikazuje, kako se lahko ključni elementi standardov ISO 9001 (kakovost), ISO 27001 (informacijska varnost) in ISO 22301 (neprekinjeno poslovanje) vključijo v posamezne faze projektnega cikla. Namen predstavljenega modela je podpreti sistematično in skladno upravljanje kakovosti, varnosti ter neprekinjenega delovanja v projektih.

Glede na predlagani model je za integracijo standardov ISO 9001, 27001 in 22301 v projektno vodenje smiselna uporaba procesno usmerjenega, na tveganjih temelječega in deležnikom prilagojenega pristopa, z vključeno podporo projektne pisarne (PMO) in postopnim prehodom v integriran sistem vodenja projektov, ki podpira zahteve vseh treh standardov.

4 RAZPRAVA

Predlagani model prispeva k boljšemu razumevanju, kako lahko organizacije sistematično pristopijo k integraciji standardov ISO 9001, ISO 27001 in ISO 22301 v okviru projektnega vodenja, zlasti na področju informacijske tehnologije. Model ponuja strukturiran okvir, ki združuje ključne vidike kakovosti, informacijske varnosti in neprekinjenega poslovanja, ter jih umešča v posamezne faze projektnega cikla. Na ta način ne omogoča le praktične uporabe v organizacijskem kontekstu, temveč tudi oblikuje osnovo za nadaljnje empirične raziskave in testiranje hipotez v različnih sektorjih.

Tabela 2: **Model integracije ISO standardov v projektno vodenje**

Faza projekta	Zahteve ISO 9001 – kakovost vodenja	Zahteve ISO 27001 – informacijska varnost	Zahteve ISO 22301 – neprekinjeno poslovanje
1. Začetek (Initiating)	Opredelitev zahtev kupcev in deležnikov	Identifikacija občutljivih informacij, dodelitev varnostnih vlog.	Preliminarna analiza vpliva motenj (BIA).
2. Načrtovanje (Planning)	Določitev ciljev kakovosti, načrt preverjanja kakovosti	Ocena varnostnih tveganj povezanih z obsegom projekta, vključitev varnostnih kontrol v projektni načrt.	Identifikacija ključnih virov in storitev, potrebnih za kontinuiteto projekta, načrtovanje ukrepov za odpornost in obnovo po motnjah.
3. Izvedba (Executing)	Validacija izhodov, spremljanje zadovoljstva uporabnika	Implementacija kontrol, sledenje spremembam, uporaba protokolov ob incidentih.	Izvedba vaj ali simulacij, zagotavljanje kritičnih funkcij.
4. Nadzor in upravljanje tveganj (Monitoring & Controlling)	Merjenje in analiza uspešnosti (KPI), upravljanje neskladij	Redno preverjanje skladnosti, zaznavanje in poročanje varnostnih incidentov, uporaba revizij in notranjih pregledov.	Spremljanje kazalnikov odpornosti, prilagajanje načrta glede na spremembe okolja ali groženj.
5. Zaključek (Closing)	Ocena zadovoljstva deležnikov, priporočila za izboljšave	Pregled skladnosti, posodobitev registra tveganj, analiza varnostnih dogodkov v okviru projekta.	Pregled, ali so bili zagotovljeni pogoji za neprekinjeno delovanje, zajetje povratne informacije.

Čeprav organizacije v praksi pogosto uvajajo več ISO standardov hkrati – kot odziv na zahteve glede varnosti, kakovosti in neprekinjenega poslovanja – se zdi, da integracija teh standardov v projektno vodenje še ni bila sistematično raziskana. Večina znanstvene literature se še vedno osredotoča na posamezne standarde in njihov vpliv na organizacijsko uspešnost, medtem ko je vpliv sočasne uporabe ISO 9001, ISO 27001 in ISO 22301 v okviru projektnega cikla še vedno premalo raziskan.

Ispas in Mironeasa [27], ugotavljata prav to, integracija standardov v t. i. integriran sistem vodenja omogoča boljše upravljanje tveganj, hitrejše doseganje strateških ciljev in boljše usklajevanje z zakonodajnimi zahtevami. Tak sistem ne obravnava standardov kot izolirane entitete, temveč kot povezana orodja znotraj širšega organizacijskega in projektnega konteksta.

Zlasti v IT sektorju, kjer so projekti dinamični in izpostavljeni stalnim motnjam, se celovita uporaba standardov lahko izkaže kot ključna. Vendar trenutna literatura le redko ponuja konkretne modele, ki bi standarde umeščali v posamezne faze projektnega vodenja (npr. PMBOK). Predlagani model integracije ISO 9001, 27001 in 22301 po fazah projektnega cikla tako predstavlja prispevek k zapolnitvi te raziskovalne vrzeli.

Predstavljeni model nam omogoča strukturirano obravnavo treh ključnih dimenzij projektnega vodenja: kakovosti, varnosti in odpornosti. Na primer, že v fazi načrtovanja se z vključitvijo ciljev kakovosti (ISO 9001), analize varnostnih tveganj (ISO 27001) in načrtovanja za neprekinjeno delovanje (ISO 22301) bistveno poveča robustnost projektnega načrta ter pripravljenost na nepričakovane dogodke. Takšen integriran pristop prispeva tudi k boljši notranji usklajenosti projektnih ekip, večji preglednosti izvajanja ter učinkovitejšemu nadzoru in poročanju skozi celoten projektni cikel.

5 SKLEP

Integracija standardov ISO 9001, ISO 27001 in ISO 22301 v projektno vodenje se zdi obetaven strateški pristop za organizacije, ki delujejo v kompleksnih in hitro spreminjajočih se IT okoljih. Predpostavlja se, da bi tak pristop lahko prispeval k poenotenju ključnih zahtev glede kakovosti, informacijske varnosti in neprekinjenega poslovanja, ter potencialno izboljšal načrtovanje, izvajanje in nadzor nad projektnimi aktivnostmi.

Na podlagi izhodiščne analize sklepamo, da bi tovrstna integracija lahko prispevala tudi k večji odpornosti na tehnološke in organizacijske motnje, saj vključuje sistematično obvladovanje tveganj in pripravljenost na nepredvidene dogodke že v zgodnjih fazah projekta. Prav tako bi usklajeno delovanje po teh treh standardih lahko okrepilo zaupanje deležnikov s poudarkom na transparentnosti, sledljivosti in zanesljivosti v celotnem projektnem ciklu.

Čeprav navedeni model še ni bil preverjen na konkretnih primerih iz prakse, kaže na smiselno usmeritev za organizacije, ki želijo okrepiti zrelost projektnega vodenja v skladu s PMBOK metodologijo in hkrati nasloviti zahteve regulatorjev, strank ter trga.

LITERATURA

[1] PMBOK GUIDE. (2021). The Standard for Project Management and A Guide to the Project Management Body of Knowledge. Seventh edition. *Project Management Institute.*

[2] *The ISO Survey*. ISO. (2023). https://www.iso.org/committe/54988.html. (Dostopano dne: 3. junij2025)

[3] Ifran, M., Hassan, M., & Hassan, N. (2019). The effect of project management capabilities on project success in Pakistan: An empirical investigation. *IEEE Access*, 7, 39417–39431. https://doi.org/10.1109/access.2019.2906851.

[4] Culot, G., Nassimbeni, G., Podrecca, M., & Sator, M. (2021). The ISO/IEC 27001 information security management standard: literature review and theory-based research agenda. *The TQM Journal*, 33(7), 76-105. https://doi.org/10.1108/tqm-09-2020-0202.

[5] *ISO/IEC 27001:2022*. ISO. (2022). https://www.iso.org/standard/27001. (Dostopano dne: 6. junij 2025)

[6] *The ISO survey.* ISO. (2023). https://www.iso.org/the-iso-survey.html. (Dostopano dne: 6. julij 2025)

[7] Slovenski standard sist en iso 22301:2020. (2019). Slovenski inštitut za standardizacijo.

[8] Berrichi, A., & Azarkan, Z. (2021). Business Continuity Plan facing COVID-19: From necessity to Alternatives. *International Journal of Accounting, Finance, Auditing, Management & Economics*, *2*(4), 597-617. https://doi.org/10.5281/zenodo.5149419.

[9] *ISO 9001:2025*. ISO. (2015). Quality management systems - Requirements. https://www.iso.org/standard/62085.html. (Dostopano dne: 6. julij 2025)

[10] Safder, A., & Yousaf, S. (2018). Influence of ISO 9001 certification on project management performance in software industry. *European Online Journal of Natural and Social Sciences*, *7* (3).

11] Martínez-Perales, S., Ortiz-Marcos, I., Juan Ruiz, J., & Javier Lázaro, F. (2018). Using Certification as a Tool to Develop Sustainability in Project Management. *Sustainability*, *10*. 1408 https://doi.org/10.3390/su10051408.

[12] *ISO 21502:2020.* ISO. (2020). https://www.iso.org/standard/74947.html. Dostopano dne: 8. julij 2025

[13] Powering best practice. (2025). https://www.axelos.com/certifications/propath/prince2-project-management. (Dostopano dne: 8. julij 2025)

[14] Manifesto for Agile Software Development. (2001). https://agilemanifesto.org/. (Dostopano dne: 9. julij 2025)

[15] *About Scrum.Org*. Scrum.Org, https://www.scrum.org/about. (Dostopano dne: 9. julij 2025)
[16] *What is lean?: Lean thinking.* Lean Enterprise Institute. (2023). https://www.lean.org/explore-lean/what-is-lean/. (Dostopano dne: 9. julij 2025)
[17] University, K. (2021). *The Official Guide to the kanban method.* Kanban University. https://kanban.university/kanban-guide/-. (Dostopano dne: 9. julij 2025)
[18] DeLadurantey, C. (2025). *Basic six sigma principles explained*. Six Sigma Online. https://www.sixsigmaonline.org/six-sigma-principles/. (Dostopano dne: 9. julij 2025)
[19] Atlassian. (n.d.). Waterfall methodology for project management. https://www.atlassian.com/agile/project-management/waterfall-methodology. (Dostopano dne: 8. julij 2025)
[20] Saida E. & Taibi N. (2021). ISO 9001 Quality Approach and Performance Literature Review. *European Scientific Journal, ESJ, 17*(1). 10.19044/esj.2021.v17n1p128.
[21] Kitsios, F., Chatzidimitriou, E., & Kamariotou, M. (2023). The ISO/IEC 27001 Information Security Management Standard: How to Extract Value from Data in the IT Sector. *Sustainability, 15*(7), 5828. https://doi.org/10.3390/su15075828.
[22] Mataracıoglu, T., & Ozkan, S. (2011). Analysis of the User Acceptance for Implementing ISO/IEC 27001:2005 in Turkish Public Organizations. *International Journal of Managing Information Technology, 3*(1), 1–14. https://doi.org/10.5121/ijmit.2011.3101.
[23] Esgarrancho, S., & Candido, C.J.F. (2020). Firm preparation for ISO 9001 Certification – The case of the hotel industry in Portugal. *Total Quality Management & Business Excellence, 31*(1), 23-42. https://doi.org/10.1080/14783363.2017.1404428.
[24] Ingason, H. T. (2015). Best Project Management Practices in the Implementation of an ISO 9001 Quality Management System. *Procedia - Social and Behavioral Sciences*, *194*, 192–200. https://doi.org/10.1016/j.sbspro.2015.06.133.
[25] Strelicz, A., & Bognár, F. (2020). Integrated Risk and Business Impact Analysis: A Kind of Support for ISO 22301. *European Scientific Journal ESJ, 16*(4), 1–13. https://doi.org/10.19044/esj.2020.v16n4p1.
[26] Ispas, L., Mironeasa, C., & Silvestri, A. (2025). A Study on the Emergence and Resilience of Integrated Management Systems in Organizations with an Industrial Profile in Romania. *Sustainability, 17*(6). 2401. https://doi.org10.3390/su17062401.
[27] Ispas, L. & Mironeasa, C. (2022). The Identification of Common Models Applied for the Integration of Management Systems: A Review. *Sustainability, 14*(6). 3559. https://doi.org/10.3390/su14063559.

▪

Ester Bradač je magistrirala na Univerzi v Mariboru smer Organizacija in managmeent informacijskih sistemov. Ukvarja se z vodenjem IT projektov v mednarodni organizaciji. Istočasno je tudi skrbnica organizacijskega standarda za neprekinjeno poslovanje 22301:2019 in notranja revizorka za več standardov. Aktivno pa se udejanja tudi na področju upravljanja sprememb, kjer znotraj organizacije sodeluje in usmerja projekte na tem področju.

Analiza posebnosti zagotavljanja kakovosti pri razvoju decentraliziranih aplikacij v okolju Ethereum

Mitja Gradišnik, Tina Beranič, Muhamed Turkanović
Fakulteta za elektrotehniko, računalništvo in informatiko, Univeza v Mariboru, 2000 Maribor
mitja.gradisnik@um.si, tina.beranic@um.si, muhamed.turkanovic@um.si

Izvleček

Pri razvoju decentraliziranih aplikacij (dApps) se tradicionalni razvojni procesi pogosto izkažejo za nezadostne. Tovrstne rešitve zahtevajo večji poudarek na tehničnih, varnostnih in uporabniških vidikih kakovosti aplikacij, kot smo jih sicer vajeni pri razvoju klasičnih rešitev. Ker je spreminjanje pametnih pogodb po namestitvi v omrežje verig blokov zahtevno oziroma nemogoče, sta temeljito testiranje ter presoja programske kode ključnega pomena za uspešen razvoj tovrstnih rešitev. Optimizacija stroškov goriva, nujna za izvrševanje programov v javnih omrežjih, predstavlja enega ključnih razvojnih izzivov, ki ga je potrebno ustrezno obravnavati. Poleg tega specifično okolje omrežij veriženja blokov zahteva ustrezne ukrepe za obvladovanje tveganj povezanih z ranljivostmi aplikacij in morebitnimi povezanimi finančnimi izgubami. Nespremenljivost, stroški goriva in zagotavljanje varnosti so le nekateri izmed ključnih razvojnih izzivov, ki jih je treba uspešno nasloviti pri izgradnji kakovostnih in stabilnih decentraliziranih aplikacij. Prispevek obravnava izzive, sodobne pristope in strategije razvoja decentraliziranih aplikacij ter podaja priporočila za njihov zanesljivejši in učinkovitejši razvoj, s čimer naslavlja ključne izzive uvajanja tehnologij veriženja blokov v industrijska okolja ter razvoja pametnih pogodb. Poseben poudarek je namenjen pametnim pogodbam, ki temeljijo na omrežju Ethereum.

Ključne besede: verige blokov, proces razvoja dApps, testiranje, vzdrževanje, obvladovanje stroškov transakcij, presoje

Analysis of Quality Assurance Specifics in the Development of Decentralized Applications in the Ethereum Environment

Abstract

In the development of decentralised applications, traditional development processes often prove insufficient. Such solutions require greater emphasis on the technical, security, and user experience aspects of application quality than is common in developing conventional applications. Since modifying smart contracts after deployment on a blockchain network is challenging or even impossible, thorough testing and code review are crucial for successfully developing such solutions. Optimising gas consumption, which is essential for executing programmes in public networks, represents one of the key development challenges that must be adequately addressed. Furthermore, the specific environment of blockchain networks demands appropriate measures to manage risks associated with application vulnerabilities and potential financial losses. Immutability, gas consumption, and security are critical development challenges that must be successfully tackled to build high-quality and stable decentralised applications. This paper addresses the challenges and modern approaches and strategies for dApp development. It offers recommendations for their more reliable and efficient implementation, thereby tackling key issues related to adopting blockchain technologies in industrial settings and developing smart contracts. Emphasis is placed on smart contracts based on the Ethereum network.

Keywords: Blockchain, dApps development process, testing, maintainability, cost management, audits

1 UVOD

Pospešen razvoj tehnologij veriženja blokov v zadnjih letih odpira vrata številnim inovacijam na področjih elektronskega poslovanja in shrambe podatkov ter pomembno prispeva k uvajanju novih poslovnih modelov [1]. Ključni pospeševalec teh inovacij je predvsem decentralizirana narava okolja verig blokov. Pred vpeljavo tehnologij veriženja blokov je naša družba temeljila na vzpostavljanju zaupanja med deležniki, pri čemer so ključno vlogo odigrali zaupanja vredni posredniki [2]. Tipičen primer takega posrednika so banke. Banke, kot zaupanja vreden posrednik, poskrbijo za prenos sredstev med komitenti, pri čemer morajo ti banki zaupati. Komitenti morajo banki zaupati svoja denarna sredstva ter zaupati v pravilnost izvedenih transakcij in zanesljivost njihovih zapisov, da bi omenjeni mehanizem lahko uspešno deloval.

Na tehnologijah veriženja blokov temelječe programske rešitve odpravijo potrebo po zaupanja vrednem posredniku med akterji, ki so vpleteni v elektronsko poslovanje. Zaupanje se prenese na decentralizirane sisteme same, konkretno na uporabo ustreznih kriptografskih metod in na delovanje porazdeljenega in decentraliziranega omrežja [3]. V praksi to pomeni, da tako grajene rešitve omogočajo varne in verodostojne transakcije med akterji, med katerim sicer ni potrebe po vzpostavitvi predhodnega zaupanja, za katerega bi skrbel zaupanja vreden posrednik. Tehnologije veriženja blokov ponujajo možnost izvajanja transakcij med dvema ali več anonimnimi deležniki preko mehanizma vnaprej dogovorjenih in zapisanih procedur, imenovanih pametne pogodbe [4]. Izboljšana varnost, transparentnost in decentralizacija elektronskega poslovanja so tako ključni stebri inovacij, ki jih prinaša vpeljava tehnologij veriženja blokov.

Inovatorji na področju vpeljave tehnologij veriženja blokov v poslovne procese izhajajo iz različnih gospodarskih domen, pri čemer prednjačijo področja decentraliziranih financ, zavarovalništva, dobavnih verig in logistike, zdravstva, upravljanja digitalnih identitet ter javne uprave [5], [6]. Tehnologije veriženja blokov omogočajo razvoj novega tipa programskih rešitev, ki realizirajo njihove ideje. Posledično se v zadnjem času pojavlja čedalje več aplikacij razvitih na temeljih tehnologij verig blokov, ki jih imenujemo decentralizirane aplikacije ali krajše dApps [7].

Jedro tehnologij predstavljajo linearno strukturirane verige blokov, v katere se zapisujejo stanja in transakcije izvršene med deležniki omrežja [8]. Operacije upravljanja s podatki so pri tem enosmerni proces. Omogočeno je zapisovanje oz. dodajanje zapisov, ne pa tudi njihovo spreminjanje ali brisanje. Tako zapisani podatki ostanejo v verigah blokov trajno [9]. Ker imajo podatkovni bloki omejeno kapaciteto, se zapisi razpršeni čez več podatkovnih blokov, ki so urejeni po času ustvarjanja tvorijo verige [8]. Celotna strukturo, na kateri decentralizirane aplikacije temeljijo in vanjo zapisujemo podatke, imenujemo porazdeljena knjiga transakcij (angl. distributed ledger) [10]. Posege v integriteto tako zapisanih podatkov preprečuje vrednost zgoščevalne funkcije vsebine bloka. Ta se zapiše v sledeči blok in tako ustvarja trajne povezave med podatkovnimi bloki. S tem se gradi verižna podatkovna struktura podprta z Merklovimi drevesi [11]. Nezmožnost brisanja zapisanih podatkov daje programskih produktom, ki temeljijo na verigah blokov, eno izmed ključnih lastnosti, in sicer nespremenljivost zapisov. Nespremenljivosti zapisov je ključna lastnost verig blokov, na kateri temelji varnost in zaupanje v ustvarjene zapise. Varnost zapisanih podatkov krepi odpornost na manipulacije in možnost sledenja spremembam [9].

Zaupanje v zapise krepi transparentnost in integriteto zapisanih podatkov, saj so, zaradi distribuiranega okolja omrežja verig blokov, vsi zapisi vidni vsem deležnikom. Za slednje je ključna porazdeljenost zapisov med vozlišči omrežja verig blokov. Protokoli pri tem poskrbijo za replikacijo, tako da ima vsako vozlišče v omrežju popolno kopijo podatkov, kar naredi zapise robustne in odporne na izgubo. Slednje dodatno prispeva k varnosti zapisanih podatkov. Potrjevanje stanja podatkov v porazdeljenem okolju s pomočjo algoritmov porazdeljenega soglasja omogoča decentralizacijo, saj prepreči odvisnost od enega samega strežnika ali ene centralne entitete, ki upravlja zapise [9]. Decentralizacija je še ena izmed ključnih edinstvenih lastnosti tehnologij blokov, ki omogoča dinamično vzpostavitev skupnosti akterjev brez potrebe po vzpostavitvi vrhnje avtoritete, ki to skupnost upravlja, hkrati pa onemogoča preprosto in dinamično spreminjanje kakršnih koli atributov delovanja takšnega omrežja.

Verige blokov bi imele močno omejeno uporabnost za poslovna okolja, če ne bi bile programabilne. Tekom razvoja v zadnjih letih so tehnologije veriženja

blokov prerastle platforme digitalnih valut ter vpeljale decentralizirane in zaupanja vredne programske rešitve [1]. Številne domene s pridom izkoriščajo prednosti decentralizacije, transparentnosti in varnosti zapisov ter možnost zaupanja, ki ga je mogoče vzpostaviti med sicer anonimnimi deležniki. Posledično je čedalje več aplikacij, ki temeljijo na tehnologijah porazdeljenih knjig [7], čemur se morajo ustrezno prilagoditi tudi razvijalci programske opreme. Čeprav je na voljo več platform za razvoj pametnih pogodb, se v nadaljevanju osredotočamo predvsem na Ethereum [12], ki predstavlja eno izmed vodilnih platform za razvoj pametnih pogodb [7]. S platformo imamo tudi največ praktičnih izkušenj. Ob tem je treba poudariti, da se članek osredotoča izključno na aplikacijski nivo verig blokov, medtem ko infrastrukturni nivo, ki v osnovi zagotavlja delovanje tovrstnih omrežij, ni predmet obravnave.

Namen prispevka je proučiti posebnosti decentraliziranih aplikacij, ki jih je potrebno v okviru razvojnega procesa aplikacij ustrezno nasloviti, da bi se izognili težavam s kakovostjo nastalih programskih rešitev. Cilj prispevka je identificirati izzive, ki jih prinaša razvoj decentraliziranih aplikacij, in razpoložljive metode in ukrepe, s katerimi tovrstne razvojne izzive uspešno naslavljamo. V skladu s cilji raziskave smo izpeljali dve osnovni raziskovalni vprašanji:

RV1: Katere so posebnosti decentraliziranih napram klasičnim aplikacijam?

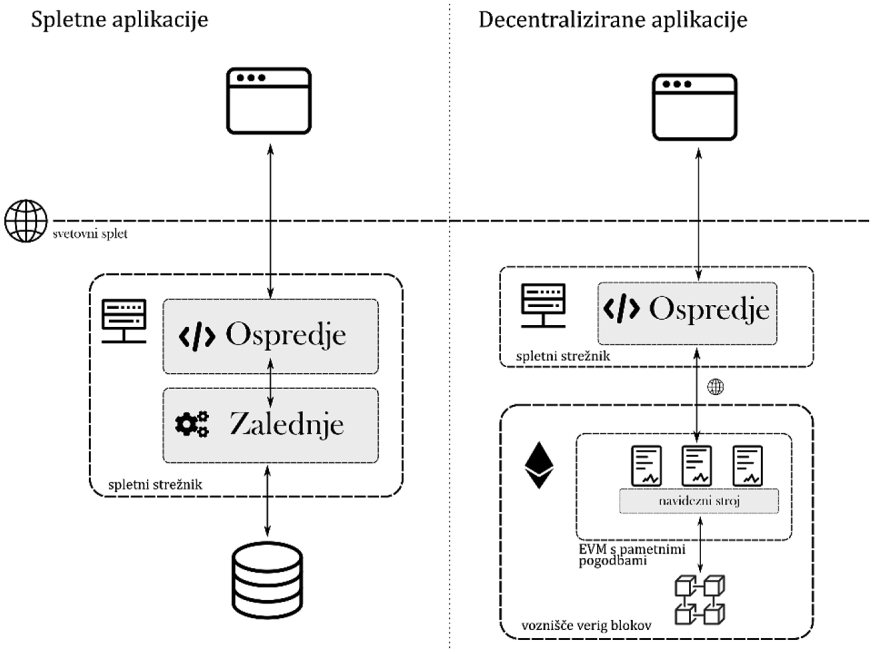
RV2: Katere razvojne metode in aktivnosti naslavljajo njihove posebnosti pri zagotavljanju kakovosti decentraliziranih aplikacij?

Raziskava se metodološko opira predvsem na pregled literature, pri čemer je bila pregledana relevantna aktualna literatura domačih in tujih avtorjev s področja programskega inženirstva. Za opazovanje, opisovanje in analiziranje identificiranih razvojnih izzivov in njihovih rešitev, bo v prispevku uporabljena metoda deskripcije. Metodološko je prispevek teoretične narave in služi kot uvod v nadaljnje raziskovalno delo na področju obvladovanja kakovosti pri razvoju decentraliziranih aplikacij. Poglavitni cilj raziskav je identificirati tiste posebnosti tehnologij veriženja blokov, ki jih je potrebno nasloviti pri gradnji kakovostnih decentraliziranih aplikacij.

2 DECENTRALIZIRANE APLIKACIJE

Arhitekturna zasnova decentraliziranih aplikacij je sicer podobna spletnim rešitvam, vendar z nekaterimi pomembnimi razlikami. V grobem so decentralizirane aplikacije sestavljene iz dveh ključnih komponent: ospredja z uporabniškim vmesnikom in zalednega dela s poslovno logiko aplikacije. Ospredje poskrbi za interakcijo uporabnikov z aplikacijo in je v veliki večini primerov realizirano kot spletna aplikacija. Poslovna logika v zaledju decentraliziranih aplikacij je običajno zapisana v pametnih pogodbah, ki predstavljajo jedro vsake decentralizirane aplikacije. Čeprav obstaja široka paleta različnih platform verig blokov s podporo pametnim pogodbam, se v praksi najpogosteje uporablja platforma Ethereum, pri čemer je za razvoj pametnih pogodb uporabljen programski jezik Solidity [7]. Pametne pogodbe implementirajo tako poslovno logiko aplikacije in hkrati poskrbijo za hrambo zapisov v verigah blokov [13]. Zaledni del aplikacije s poslovno logiko in podatkovna baza za hrambo podatkov pri decentraliziranih aplikacijah nista več potrebna, vendar sta še vedno lahko prisotna. Kadar sta prisotna, govorimo o hibridnih arhitekturah, kjer se del podatkov hrani v podatkovno bazo, del pa se jih zapiše v verige blokov [14]. Primerjavo med arhitekturama klasičnih spletnih aplikacij in decentraliziranih aplikacij [15] prikazuje Slika 1. Iz arhitekturne sheme je razvidno, da je ospredje decentraliziranih aplikacij enako ospredju spletnih aplikacij in se z razvojnega vidika ne razlikujeta veliko. Posebnosti razvoja decentraliziranih aplikacij izhajajo iz zaledja s pametnimi pogodbami. Večina posebnosti v razvoju decentraliziranih aplikacij izhaja ravno iz zalednega dela aplikacije in dejstva, da mora biti ospredje primerno usklajeno z zaledjem, ki je temelječe na pametnih pogodbah, kakor tudi povezljivo z uporabniškimi denarnicami tehnologije veriženja blokov. Pri vsem tem nam pomagajo namenske knjižnice, kot sta web3 [16] ali ethers [17]. Upoštevati je potrebno dodatno arhitekturno dejstvo. In sicer, v primeru potrebe po shranjevanju večjih količin podatkov ali datotek, se, z razliko od uporabe centraliziranih strežnikov, v dApp poslužujemo decentraliziranih sistemov shranjevanja datotek, kot je IPFS.

Osrednji del decentraliziranih aplikacij predstavljajo pametne pogodbe. Te sledijo v devetdesetih letih predstavljeni ideji samoizvršljivih pogodb, ki temeljijo na transakcijskem protokolu, ki ga avtono-



Slika 1 **Primerjava arhitekture spletnih in decentraliziranih aplikacij**

mno izvaja računalnik [4]. V osnovi gre za samoizvršljivo programsko kodo, ki se avtonomno izvede, kadar so za to izpolnjeni vnaprej definirani pogoji [18]. Pametne pogodbe bi lahko tako primerjali s konceptom klasičnih pogodb iz realnega sveta, pri čemer so pametne pogodbe povsem digitalizirane. Kot take so shranjene na vozliščih omrežja verig blokov.

2.1 Okolje izvajanja decentraliziranih aplikacij

Pomembna razlika med spletnimi in decentraliziranimi aplikacijami izhaja iz izvajalnega okolja, v katerem aplikacije tečejo. Spletne aplikacije običajno tečejo na spletnih strežnikih, ki zagotovi izvajalno okolje za njihovo izvajanje. Znotraj spletnega strežnika običajno teče celotna spletna aplikacija. Od aplikacije je običajno oddvojena le podatkovna baza. Nekatere zadolžitve prevzame sam spletni strežnik, in sicer skrb za usmerjanje prometa do aplikacije in predhodna obdelava HTTP zahtev, varnost aplikacij, upravljanje uporabnikov in dostopov ter poganjanje spletnih aplikacij [19]. Upoštevati je potrebno tudi skaliranje aplikacij, ki zajema tudi horizontalno skaliranje zalednih delov in samih spletnih strežnikov. Velikokrat se poslužujemo za ta namen avtomatiziranih pristopov, ki nam jih ponujajo ponudniki oblačnih storitev. Decentralizirane aplikacije močno zmanjšajo odvisnost od spletnih strežnikov, saj jih uporabljajo zgolj za poganjanje ospredja z uporabniškim vmesnikom. Pametne pogodbe, kot zaledna komponenta decentraliziranih aplikacij, tečejo ločeno od ospredja na vozliščih omrežja verig blokov v navideznih strojih sistemov v omrežju. Na primer, pametne pogodbe napisane za platformo Ethereum se izvajajo v Ethereum navideznih strojih [7]. Upoštevajoč zapisano, lahko naredimo primerjavo z oblačno skalirano infrastrukturo, vendar je ključna razlika ponovno decentralizacija, saj se v primeru klasične oblačne infrastrukture vsi procesi izvajajo in nadzorujejo s strani centralnega ponudnika takšnih storitev. Prav tako je okvir funkcionalnosti pri navideznih strojih v verigah blokov močno okrnjen, saj ti, za razliko od spletnih strežnikov, skrbijo izključno za izvajanje programske kode pametnih pogodb. Pomembna razlika okolij pametnih pogodb v primerjavi s vsebnikom spletnih strežnikov je v načinu zaračunavanja stroškov. Ker tečejo pametne pogodbe porazdeljeno po vozliščih omrežja verig blokov, je potrebno za uporabo infrastrukture plačati pristojbine za uporabo. Decentralizirane aplikacije so torej močno občutljive na količine podatkov, katere obdelujejo in zapisujejo.

3 POSEBNOSTI RAZVOJA DECENTRALIZIRANIH APLIKACIJ

Primerjava arhitekture decentraliziranih aplikacij s sorodnimi spletnimi rešitvami nakazuje na večjo kompleksnost decentraliziranih aplikacij [13]. V

praksi se izkaže, da je decentralizirane aplikacije posledično tudi težje razvijati. Arhitektura decentraliziranih aplikacij, posebnosti pametnih pogodb in lastnosti omrežij verig blokov torej zahtevajo prilagojene procese razvoja tovrstnih programskih rešitev.

3.1 Nespremenljivost pametnih pogodb

Spremembe so neizogiben del življenjskega cikla programskih produktov. Redki so programski produkti, ki po izdajni različici ne bi deležni nobenih sprememb ali popravkov. Ne glede na to, kako dobro so bile v začetni fazi razvoja programskega produkta zajete zahteve uporabnikov in kako dobro so bile kasneje v fazah razvoja te prenesene v programski produkt, sčasoma postaja razkorak med implementirano programsko rešitvijo in pričakovanji uporabnikov čedalje večji. Vzrokov za omenjen pojav je več. Na primer, delovni procesi v organizacijah, v okviru katerih se programska oprema uporablja, se neprestano spreminjajo in nadgrajujejo. Gonilo sprememb so bodisi nadgradnje bodisi razširitve delovnih procesov. Pričakovano je, da tem spremembam sledi tudi programska oprema in se na spremembe ustrezno prilagodi. Četudi bi delovni procesi ostali nespremenjeni, na spremembe programske opreme napeljuje kup zunanjih dejavnikov, kot so spremembe v zakonodaji, spremembe povezanih sistemov in spremenjeno okolje delovanja organizacije. Nenazadnje je potrebno upoštevati, da se običajno tekom uporabe programskega produkta odkrije nekaj hroščev, ki se jih v fazi testiranja ni zaznalo in jih je potrebno za nemoteno uporabo aplikacije odpraviti. Spremembe so torej sestavni del življenjskega cikla programskega produkta, ki ga sodobni razvojni procesi uspešno naslovijo.

Glede na sorodnost in podobnost med decentraliziranimi in spletnimi aplikacijami, bi se na prvi pogled zdelo smiselno, da se dobro preizkušeni in uveljavljeni agilni proces razvoja spletnih rešitev uporabi tudi pri razvoju decentraliziranih aplikacij. Ker pametnih pogodb, kot osrednje komponente decentraliziranih aplikacij, ko so te enkrat nameščene v okolje verig blokov, ne moremo enostavno spreminjati, decentralizirane aplikacije ne zahtevajo faze vzdrževanja v klasičnem pomenu besede [20]. Izkaže se, da agilnega procesa razvoja spletnih rešitev ni mogoče enostavno in brez prilagoditev prenesti v okvir razvoja decentraliziranih aplikacij.

Kratki iterativni razvojni cikli agilnih pristopov poskrbijo za odzivnost na spremembe v uporabniških zahtevah, s tem da prenesejo te spremembe v čim krajšem času v programski produkt. Pristopi in avtomatizacija procesa, ki jo vpelje DevOps poskrbijo, da je razvojni proces pri tem čim bolj učinkovit. Rezultat iterativnih razvojnih ciklov so torej vedno nove posodobljene različice programskih rešitev, ki se preko različnih mehanizmov avtomatizacije razvojnega procesa v čim krajšem možnem času prenesejo v produkcijsko okolje in predajo v uporabo. Naveden proces se uporablja pri različnih tipih programskih produktov, ne glede na platformo, programski jezik ali namen uporabe.

Čeprav za širok spekter programskih produktov opisan pristop v praksi deluje odlično in se izkaže za izjemno učinkovitega, tovrstni pristop k razvoju pri decentraliziranih aplikacijah trči na oviro. Če so običajne aplikacije odlično prilagojene neprestanemu spreminjanju in nadgrajevanju, so spremembe decentraliziranih aplikacij trši oreh. Pametnih pogodb, kot osrednje komponente decentraliziranih aplikacij, zaradi lastnosti tehnologij veriženja blokov ni mogoče enostavno spreminjati, ko so te enkrat nameščene v omrežje verig blokov. Navedeni varnostni mehanizem pametnih pogodb, ki preprečuje možnost njihovega kasnejšega zlonamernega spreminjanja, postavlja dodatne izzive pri procesu njihovega dolgoročnega vzdrževanja. V praksi to pomeni, da tudi manjše nadgradnje ali popravki hroščev v algoritmu pametnih pogodb, rezultirajo v ponovno nameščanje nove različice pametne pogodbe, ki prejšnjo nadomešča, kar povzroči izgubo podatkov, vezanih na predhodno različico. Razvojni cikel decentraliziranih aplikacij mora slediti načelu, da je potrebno pametne pogodbe spisati s popolnim naborom funkcionalnosti in vsa v teoriji brez programskih hroščev, zaradi katerih bi bili kasneje prisiljeni v izdajo posodobljenih različic. Ker pa bo do sprememb zagotovo prišlo, uporabljajo razvijalci decentraliziranih aplikacij različne nadgradljive arhitekture, da omogočijo posodabljanje funkcionalnosti. Najpogostejši pristop je uporaba **vzorca posrednika** [21], [22], kjer glavna pogodba preusmerja klice na ločeno, zamenljivo pogodbo z dejansko logiko. Naprednejši, a bolj kompleksen je **diamantni vzorec** [23], ki omogoča modularne nadgradnje posameznih funkcionalnosti. Alternativno se včasih uporabi preprosta **migracija na novo pogodbo**, kjer se decentralizirana aplikacija poveže na nov naslov, stanje pa se prenese ročno ali s skriptami. Dodatno fleksibilnost ponuja **vzorec register** [21],

kjer osrednja pogodba hrani naslove vseh aktivnih komponent. Izbira pravega mehanizma je odvisna od kompleksnosti sistema, potrebe po fleksibilnosti ter varnostnih in uporabniških zahtev.

3.2 Izpostavljenost med izvajanjem

Spletne aplikacije ne tečejo kot samostojni procesi na sistemu, temveč so nameščene v vsebnik spletnega strežnika. Spletni strežnik spletnim aplikacijam nudi varno izvajalno okolje in jih neposredno zaščiti pred zunanjim okoljem. Kadar poteka komunikacija s spletno rešitvijo, zahtevke sprejme in predhodno obdela spletni strežnik. Šele po predobdelavi so zahtevki posredovani spletnim aplikacijam. Spletni strežnik je torej zadolžen za kopico funkcionalnosti, performančnih in varnostih mehanizmov spletnih aplikacij, za katere razvijalcem spletnih rešitev ni potrebno neposredno skrbeti. Običajno je potrebna zgolj ustrezna konfiguracija spletnega strežnika.

Za razliko od spletnega strežnika, navidezni stroj na vozliščih omrežij verig blokov pametnim pogodbam ne nudi tako širokega nabora funkcionalnosti, temveč zgolj izvajalno okolje pametnih pogodb. Pametne pogodbe v okolju omrežja verig blokov delujejo kot povsem samostojne aplikacije, ki same sprejemajo zahtevke uporabnikov, jih izvršijo in klicatelju same vrnejo rezultat obdelave. Navidezni stroj pri tem ne nudi nobenih dodatnih funkcionalnosti, niti ne poskrbi za performančne in varnostne vidike decentraliziranih aplikacij, razen preverjanja celovitosti transakcij. Na razvijalcih pametnih pogodb je torej, da sami v programski kodi pametne pogodbe poskrbijo, da je izvajanje pogodbe performančno učinkovito, varno ter dostopno zgolj uporabnikom, ki do izbranih funkcionalnosti pametnih pogodb smejo dostopati. Razvijalci pametnih pogodb tudi nosijo breme odgovornosti, da je implementacija pametnih pogodb brezhibna in ne predstavlja nepotrebnega varnostnega tveganja za uporabnike decentraliziranih aplikacij.

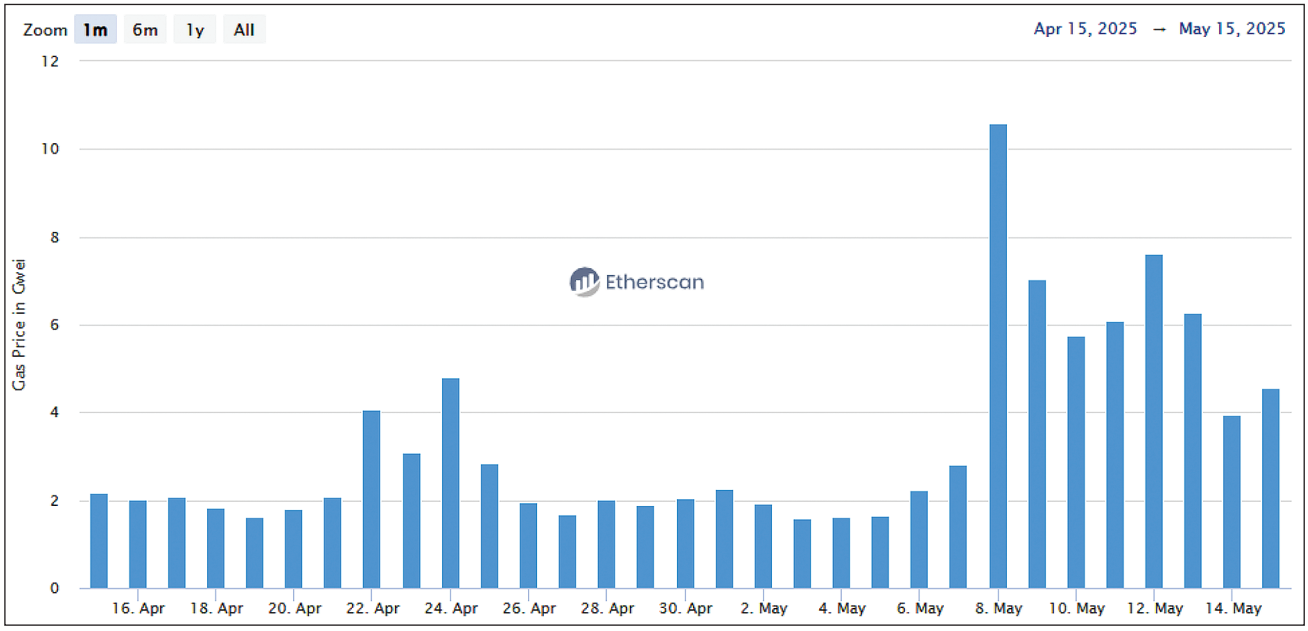
3.3 Obvladovanje stroškov uporabe infrastrukture

Performančna učinkovitost je pomemben atribut kakovosti programskih komponent, ki ga je potrebno ustrezno nasloviti pri razvoju programskih rešitev, ne glede na njihovo vrsto. Performančno neučinkovita programska oprema po nepotrebnem porablja sistemske vire in vpliva na stroške uporabljene infrastrukture ne glede na to, ali je ta najeta ali lastniška. Poudarek na stroškovni učinkovitosti je pri pametnih pogodbah še posebej izrazit.

Pametne pogodbe, nameščene na javnih verigah blokov, kot je Ethereum, se izvajajo na vozliščih, pri čemer je treba za uporabo infrastrukture plačati pristojbine za gorivo (angl. gas fees) [7]. Po prehodu Ethereuma na mehanizem soglasja ***Proof of Stake*** (PoS) so te pristojbine še vedno prisotne, čeprav se sedaj plačujejo overjevalcem (angl. validators) namesto rudarjem [24]. Cena pristojbine v posameznem omrežju je odvisna od kompleksnosti pametne pogodbe, zasedenosti omrežja in cene domorodne kriptovalute, v kateri se gorivo zaračunava [7], [25]. Gibanje cene pristojbine za omrežje Ethereum, izraženo v enoti GWei [7], za obdobje med 15. aprilom in 15. majem 2025 prikazuje Slika 2. GWei lahko razumemo kot manjšo enoto kripto kovanca, ki meri porabljeno računsko moč, potrebno za izvedbo operacij nad verigami blokov. Operacije na verigami blokov sicer definira algoritem zapisan v obliki pametnih pogodb. Temeljne operacije nad verigami blokov, za katere je potrebno plačevati pristojbine v omrežju Ethereum, na primer vključujejo zapisovanje podatkov v shrambo in računske operacije nad podatki. Pristojbine za gorivo so prav tako pomembne za ustvarjanje novih pogodb in pošiljanje žetonov. V omrežju verig blokov se izvajanje pametnih pogodb zaračunava po izvedenih operacijah zapisovanja podatkov. Praksa kaže, da stroški izvajanja pametnih pogodb v javnih omrežjih hitro narastejo [13], zaradi česar je njihovo obvladovanje ključno za vzdržnost dolgoročnega delovanja decentraliziranih aplikacij. V kolikor se za izvajanje decentraliziranih aplikacij uporabljajo javna omrežja, je pri njihovem razvoju torej potrebno vnaprej natančno proučiti predvidene stroške izvajanja pametnih pogodb in po potrebi izvesti njihovo optimizacijo.

3.4 Odprtokodnost programske kode

Javna objava programske kode pametnih pogodb je dobra praksa, ki ima številne pozitivne učinke. Odprtost kode zagotavlja transparentnost in prispeva k zaupanju med deležniki v omrežju. Vsak izmed deležnikov ima tako možnost vpogleda v algoritem pametne pogodbe in možnost preučitve njenega natančnega delovanja. S tem se lahko prepriča, da se pametna pogodba izvaja na pričakovan način. Transparentnost izvajanja pametnih pogodb je ključni mehanizem za preprečevanje morebitnih zlorab,



Slika 2: **Gibanje cene goriva za omrežje Ethereum [26]**

goljufij in manipulacij, kar pomembno pripomore k zaupanju v decentralizirane aplikacije [5]. Dodatno daje odprtokodnost priložnost vsem članom skupnosti, da prispevajo svoj delež k izboljšavam programske kode, na primer optimizaciji delovanja, odpravo najdenih napak ali dopolnitev funkcionalnosti. Pomemben vidik odprtokodnosti predstavlja tudi varnostna analiza pametnih pogodb, ki jo lahko opravi snovalec decentralizirane aplikacije ali skupnost.

Možnost vpogleda v izvorno kodo pametnih pogodb po drugi strani odpre vrata tveganjem, povezanim z izkoriščanjem opaženih pomanjkljivosti v programski kodi. Programska koda lahko vsebuje nezakrpane varnostne luknje, napake v logiki ali nepreverjene vhodne podatke. Te se v programski kodi kažejo v obliki logičnih napak ali uporabe slabih praks programiranja. Na tveganja, povezana z javno izpostavljenostjo izvorne kode pametnih pogodb, dodatno vpliva okolje, v katero so pametne pogodbe nameščene in na katerem se izvajajo. Navidezni stroji pametnim pogodbam ne nudijo nobenega dodatnega varnostnega ovoja ter jih navzven neposredno izpostavljajo. Pametne pogodbe lahko v javnih omrežjih pokliče kdorkoli, upoštevajoč vse morebitne implementirane modifikatorje in specifikacije vidnosti. Zato je toliko bolj pomembno, da je njihova programska koda brez morebitnih programskih pomanjkljivosti, saj so sicer nevarne za uporabo.

4 PRILAGODITVE RAZVOJNEGA PROCESA DECENTRALIZIRANIH APLIKACIJ

Nezmožnost enostavnega spreminjanja pametnih pogodb po namestitvi v okolje verig blokov, velika občutljivost, povezana z varnostnimi tveganji in relativno veliki stroški izvedenih operacij v omrežju nakazujejo na potrebo po prilagojenem razvojnem procesu. Sodobni agilni razvojni procesi s hitrim odzivanjem na uporabniške zahteve in avtomatizirano neprekinjeno integracijo in dostavo vedno novih različic programskega produkta se pri razvoju decentraliziranih aplikacij ne izkažejo za najučinkovitejši pristop k razvoju. Za kakovosten in stroškovno učinkovit razvoj pametnih pogodb je torej ključno, da se v fazi razvoja, pred namestitvijo v vozlišča verig blokov, zagotovi, da so te ustrezno načrtovane ter da ne vsebujejo programskih hroščev [7], ki bi razvijalce silile v izdajo popravljenih različic aplikacij.

4.1 Ključnost testiranja v zgodnjih fazah razvoja

Ker pametnih pogodb ni mogoče enostavno nadgrajevati, kot smo to sicer vajeni pri drugih družinah aplikacij, se je potrebno pred namestitvijo rešitve prepričati, da je rešitev funkcionalno ustrezna in ne vsebuje programskih hroščev, zaradi katerih bi bili prisiljeni v izdajo popravljene različice. V fazi načrtovanja je potrebno skrbno in celostno načrtovati tako funkcionalne kot arhitekturne vidike aplikacij. Prvi

v vrsti ukrepov za zagotavljanje kakovosti decentraliziranih aplikacij je obsežno in celovito testiranje. Sistematično testiranje v zgodnji fazah zagotovi, da se načrtovalske in implementacijske napake zaznajo in odpravijo še znotraj razvojnega cikla pred namestitvijo v produkcijsko okolje in predajo v uporabo. V praksi se izkaže, da zna biti odloženo odpravljanje programskih hroščev v kasnejših fazah stroškovno drago [27]. Praksa odloženega razreševanja razvojnih težav (angl. Delayed Issue Effect) sicer velja za slabo prakso tudi pri razvoju drugih družin aplikacij, vendar je težavnost odpravljanja napak zaradi nespremenljivosti pametnih pogodb po namestitvi na vozlišča verig blokov pri decentraliziranih aplikacijah toliko večja in ji je potrebno posvetiti več pozornosti [28]. Da bi preprečili nepotrebne stroške in škodo povezano z izgubo podatkov, se je potrebno izogniti napakam v produkcijskih različicah programskih rešitev.

Kljub temu, da se za razvoj pametnih pogodb uporabljajo manj uveljavljeni programski jeziki in platforme, ki so na voljo krajši čas in se uporabljajo v ožjem krogu razvijalcev, imajo razvijalci pametnih pogodb na voljo kopico testnih ogrodij in orodij, s katerimi je mogoče izvesti sistematično in temeljito testiranje. Slednje temelji na doslednem testiranju enot (angl. unit testing) in izvedbi testiranja fuzz. Pri testiranju enot se večinoma preverja pravilnost izračunov, delovanje nadzora dostopa in dovoljenja, upravljanje z dogodki itd. Testi fuzz predstavljajo tehniko avtomatiziranega testiranja, pri katerem se v vhode programov vnašajo obsežni, običajno naključno generirani in nepričakovani vhodni podatki, namen katerih je preveriti ustreznost delovanja aplikacije pod različnimi pogoji. Foundry [29] je primer orodja, ki omogoča takšno vrsto testiranja, pri čemer se lahko uporablja tudi za testiranje enot. Za slednje se sicer uporablja tudi Hardhat [30]. Del testiranja je lahko tudi avtomatski pregled izvorne kode za znane ranljivosti (npr. ponovni vstop, prelivanje, izvor transakcije, posredni klic). Tudi sicer je za testiranje decentraliziranih aplikacij potrebno ubrati učinkovitejši pristop, saj je strošek testiranja zaradi visokih stroškov decentraliziranega okolja bistveno višji, kot je strošek testiranja drugih družin aplikacij [5].

4.2 Testiranje porabe goriva

Izbira podatkovnih tipov in struktur podatkov, števila izvedenih ciklov in nabora ukazov, vrstnega reda inicializacij in ovrednotenj, ostaja povsem suverena odločitev razvijalcev pametnih pogodb [7]. Ne glede na razvijalčevo izbiro so nekateri zapisi algoritmov iz performančnega vidika učinkovitejši kot drugi, kar lahko pri izvajanju pametnih pogodb povzroči razlike v porabi goriva. Poglavitni cilj testiranja porabe goriva je preveriti, ali je poraba goriva izvajanja pametnih pogodb smotrna oz. ali v programski kodi pametnih pogodb obstajajo segmenti, ki bi jih bilo mogoče izvesti učinkoviteje. Naloga performančnega testiranja je torej preveriti, da programska koda na primer ne vsebuje nepotrebne programske kode, ne izvaja nepotrebnih iteracij zank in po nepotrebnem ne troši prostora za shranjevanje podatkov.

K testiranju porabe goriva pametnih pogodb pristopimo z dinamičnim testiranjem obnašanja aplikacije in statično analizo programske kode. Oceno porabe goriva lahko pridobimo z namestitvijo pametnih pogodb v simulirana omrežja, ki so lahko nameščena na lokalnem razvojnem ali zasebnem strežniku. Poganjanje in testiranje pametnih pogodb v lokalnem okolju omogoča oceno o sprejemljivosti stroškov izvajanja pametnih pogodb. Oceno stroškov porabe goriva pametnih pogodb je seveda mogoče izvesti neposredno na javnih omrežjih [31], vendar v tem primeru že nastanejo dejanski stroški. Če poraba goriva pri testiranju v simuliranem okolju ni zadovoljiva, je potrebno programsko kodo pametnih pogodb stroškovno optimizirati.

Stroškovna optimizacija programske kode pametnih pogodb temelji na statični analizi programske kode. Analiza zajema prepoznavo segmentov programske kode, ki veljajo za potratne. Takšne segmente programske kode označujemo kot smrdečo kodo. Sledi odstranitev nepotrebnih ali minimiziranje potratnih operacij na najmanjši možni obseg, potreben za realizacijo funkcionalnosti pametne pogodbe. V programski kodi se tako išče smrdečo kodo, na primer [7]:

- Podvojeni zapisi, kot so nepotrebna posodabljanja vrednosti spremenljivk znotraj zank.
- Neskončne zanke.
- Neučinkovitost pri inicializaciji spremenljivk, predvsem njihova inicializacija na njihove privzete vrednosti (npr. inicializacija tipa uint256 na vrednost 0).
- Neučinkovita raba podatkovnih tipov (če je le mogoče, je uporaba bytes32 bolj učinkovita od podatkovnega tipa niz).

- Neučinkovita uporaba indeksiranih parametrov.
- Neučinkovito delo z zunanjimi funkcijami.

Optimizirano različico pametnih pogodb se ponovno namesti v simulirano omrežje, kjer se ponovno oceni stroške izvajanja. Postopek se ponavlja, dokler rezultat stroškovne optimizacije ni zadovoljiv [7].

Identifikacija smrdeče kode preko analize programske kode velja za kompleksen in dolgotrajen postopek, ki terja za uspešno izvedbo opravila izkušenega strokovnjaka. Postopek identifikacije smrdeče kode pametnih pogodb, ki po nepotrebnem povečujejo stroške izvajanja, močno pospešijo orodja za optimizacijo porabe. Čeprav je ročno preverjanje kode mogoče, so avtomatizirana orodja pri tem učinkovitejša [32]. Orodje GasMet [7] opravi analizo programske kode pametnih pogodb napisanih v programskem jeziku Solidity, pri tem pa je sposobno zaznati 19 vzorcev, ki po nepotrebnem prispevajo k večji porabi goriva. Analiza, opravljena z orodjem, daje razvijalcem jasne indice, kako se lotiti optimizacije porabe pametnih pogodb. Na voljo sta tudi prototipa orodij GasChecker [33] in Gasper [34] , ki analizo pametnih pogodb opravita s simbolično izvedbo na nivoju zlogovne kode pametnih pogodb. Orodje GasChecker prepozna deset neučinkovitih programski vzorcev, Gasper jih prepozna sedem. Prav tako lahko za oceno stroškov goriva uporabljamo že omenjena orodja za testiranje, kot sta Hardhat in Foundry. Ne glede na pristop analize pametnih pogodb, orodja zgolj zaznajo potencialno problematično programsko kodo. Končna odločitev o ukrepih glede optimizacije porabe ostaja v pristojnosti razvijalcev.

4.3 Preverjanje varnosti

Pomemben vidik zagotavljanja kakovosti decentraliziranih aplikacij je tudi zagotavljanje varnosti pametnih pogodb. Ker so pametne pogodbe programske komponente, ki so nameščene na javno omrežje v verigah blokov in tako javno izpostavljene morebitnim napadalcem, je pravočasno odkrivanje ranljivosti ter preprečevanje vdorov nujno za njihovo dolgoročno in nemoteno delovanje.

Cilj varnostne analize je preveriti, ali so v programski kodi pametnih pogodb prisotne varnostne ranljivosti, ki predstavljajo tveganja za njihovo delovanje in bi lahko ogrozile dolgoročno vzdržnost izvajanja decentraliziranih aplikacij. Podobno kot optimizacija porabe pametnih pogodb, se tudi analiza varnostnih ranljivosti izvede s statično analizo programske kode in dinamično analizo obnašanja aplikacije med izvajanjem. V praksi je bilo prepoznanih več vzorcev, ki lahko ogrozijo nemoteno delovanje decentraliziranih aplikacij. Vidnejši med njimi so [8]:

- Ponovni vstop iste funkcije omogoča večkratno izvajanje, kar lahko pripelje do neveljavnih stanj podatkov.
- Neustrezna avtentikacija.
- Nepooblaščeno samouničenje pametnih pogodb-.
- Nezaščitene funkcije.
- Uporaba zastarelih funkcij.
- Nepreverjeni klici.
- Ranljivosti povezane z izgubo sredstev.

V okviru statične analize namenjene prepoznavi ranljivosti se v programski kodi pametnih pogodb identificira vzorce kodiranja, ki predstavljajo potencialne ranljivosti in bi jih napadalci lahko izkoristili v napadu [32]. Poleg prepoznavanja vzorcev povezanih z ranljivostmi, se statična analiza osredotoča na preučevanje strukture programske kode, na primer na analizo toka kontrole čez program, analizo podatkov, obravnavo napak in integracijo z uporabniškim vmesniki. Pri identifikaciji si je smiselno pomagati z orodji, ki olajšajo ročne preglede programske kode, saj jasno pokažejo na dele kode, ki prestavljajo morebiten vir varnostnih težav.

Dinamična analiza temelji na analizi programov z namenom odkrivanja ranljivosti, ki se razkrijejo šele med njihovim izvajanjem [32]. Tovrstna analiza običajno temelji na tehnikah testiranja, ki generirajo širok nabor naključno ali delno naključno ustvarjenih vhodnih podatkov [8]. Običajno se za to uporabi metoda testiranja fuzz. Generirani vhodni podatki so nato posredovani na vhode pametnih pogodb. Sledi natančno spremljanje obnašanja pametnih pogodb in detekcija morebitnega neželenega obnašanja, ki bi lahko predstavljalo varnostno tveganje, če bi se zgodilo v produkcijskem okolju med uporabo aplikacije. Takšno varnostno testiranja temelji na izvajanju pametnih pogodb v simuliranem okolju ali testnem omrežju z najrazličnejšim naborom vhodov, pri čemer se opazuje ustreznost delovanja pametnih pogodb med izvajanjem. Cilj metode je odkriti morebitne ranljivosti, izjeme ali nepričakovano obnašanje med izvajanjem. Pomemben pristop dinamične analize pametnih pogodb je prav tako analiza poti čez program pametnih pogodb. Ker temelji na dejanskem izvajanju program-

ske kode, dinamična analiza omogoča odkrivanje ranljivosti, ki so odvisne od specifičnih vhodnih podatkov, povezanih poti izvajanja in interakcij, ki nastanejo znotraj programa med delovanjem.

4.4 Revizije in presoje

Glavnina testiranja pri sodobnih pristopih razvoja programske opreme temelji na avtomatiziranih metodah. Orodja, ki jih za ta namen uporabljamo, so običajno specializirana in pokrivajo izbrane vidike kakovosti pametnih pogodb. V praksi pa se izkaže, da ta orodja ne zajemajo celotnega spektra ranljivosti in pomanjkljivosti programske kode. Pogosto zato niso sposobna zaznati tistih ranljivosti ali pomanjkljivosti, ki zahtevajo poznavanje širšega konteksta ali poglobljeno razumevanje delovanja aplikacije [5]. Presojevalci so zato pomemben steber zagotavljanja kakovosti decentraliziranih aplikacij, saj pomembno dopolnjujejo avtomatizirana orodja za testiranje in zapolnijo vrzeli, ki jih avtomatizirani postopki ne morejo pokriti, zlasti pri pregledu pametnih pogodb za prisotnost kompleksnejših ranljivosti.

Ključna razloga za izvedbo presoj programske kode pametnih pogodb sta dva. Prvič, s presojevalci se cilja predvsem na razkritje ranljivosti, ki jih orodja za avtomatizirano testiranje ne zaznajo. Presoje so pri razvoju decentraliziranih aplikacij razmeroma pogoste. Glede na študije o testiranju pametnih pogodb v odprtokodnih projektih kar 24,5 % preučenih projektov vključuje poročila o presojah, izvedenih s strani zunanjih presojevalcev [35]. Presojevalci, ki morajo biti izkušeni strokovnjaki za tehnologije veriženja blokov, izvedejo sistematičen ročni pregled programske kode, svoja dognanja strnejo v poročilu in podajo priporočila za izboljšave.

Drugi ključni razlog za izvedbo presoj izhaja iz dejstva, da pametnih pogodb ni mogoče enostavno popravljati, ko so te enkrat nameščene v omrežje verig blokov [36]. Pred namestitvijo se je namreč potrebno prepričati, da te delujejo brezhibno. Naloga presojevalcev je torej preveriti, ali pametne pogodbe izvajajo v skladu s pričakovanji uporabnikov brez nepričakovanih ali neželenih stranskih učinkov. Tovrstna presoja zajema celostno analizo ustreznosti izvajanja algoritmov, zapisanih v pogodbah, ter pravilno posodabljanje njihovih stanj. Presoje, zlasti tiste, ki jih izvedejo zunanji presojevalci, torej bistveno pripomorejo k zaupanju v kakovost implementacije pametnih pogodb.

5 SKLEP

Tehnologije veriženja blokov odpirajo možnosti inovacijam v poslovnih modelih na najrazličnejših domenah elektronskega poslovanja. Ključna prednost tehnologij veriženja blokov je možnost soustvarjanja deljenih zapisov, odpornih proti cenzuri, zlonamernim spremembam ali njihovem kasnejšem zanikanju. Z rastjo in razvojem tehnologij veriženja blokov se pojavlja potreba po programskih rešitvah, ki te ideje prenesejo v programsko kodo. V praksi se izkaže, da se pri razvoju decentraliziranih aplikacij pojavljajo posebnosti, ki jih je potrebno v okviru razvojnega procesa ustrezno nasloviti, da bi dosegli kakovostne in za uporabnika sprejemljive programske rešitve.

Posebno pozornost je potrebno nameniti aktivnostim zagotavljanja kakovosti, vključno s testiranjem. Razvijalci decentraliziranih aplikacij se v razvojnem procesu poslužujejo številnih metod in orodij, s katerimi premoščajo izzive razvoja tovrstnih aplikacij. Sistematično, temeljito in z avtomatizacijo podprto testiranje enot tekom razvoja zagotavlja, da se programska koda pametnih pogodb dodobra očisti programskih hroščev, preden je ta predana v produkcijsko okolje. Pri tem se zasleduje načelo programske kode brez hroščev. Orodja za identifikacijo slabih vzorcev in praks porabe goriva omogočajo optimizacijo programske kode pametnih pogodb, da ta pri izvajanju v javnih omrežjih porabi čim manj goriva in tako zniža operativne stroške distribuiranih aplikacij v sicer stroškovno dokaj dragem okolju. Orodja za prepoznavo varnostno ranljive programske kode predstavljajo steber zagotavljanja varnosti decentraliziranih aplikacij. Nenazadnje vseh izzivov, povezanih z zagotavljanjem kakovosti decentraliziranih aplikacij, ni mogoče nasloviti z uporabo orodij. Presoje programske kode pametnih pogodb, opravljene s strani usposobljenih strokovnjakov iz področja tehnologij veriženja blokov, dajejo dodatno zagotovilo, da so decentralizirane aplikacije grajene kakovostno in bodo v produkcijskem okolju upravičile pričakovanja uporabnikov.

Pri razvoju decentraliziranih aplikacij predstavlja uporaba sodobnih razvojnih metod, ki temeljijo na agilnih pristopih in DevOps, odprto raziskovalno področje. Agilni pristopi in DevOps temeljijo na hitrem razvoju, visoki stopnji avtomatizacije razvojnega procesa, inkrementalnem razvoju in sprotnem odzivanju na potrebe uporabnikov. Navedene lastnosti

sodobnih agilnih razvojnih procesov je pri razvoju decentraliziranih aplikacij težje doseči. Nespremenljivost pametnih pogodb, dodatna performančna testiranja in večja potreba po revizijah končnih različic programskih rešitev pred izdajami namigujejo na potrebo po prilagojenih razvojnih procesih. Ali je za razvoj kakovostnih decentraliziranih aplikacij primeren prilagojen agilni razvojni proces ali pa je v primeru razvoja tovrstnih rešitev najprimernejši slapovni procesni model, ostaja aktualna tema raziskav na tem področju. Kot prihodnje delo načrtujemo predstavitev v prispevku opisanih primerov iz prakse, kar bo dodatno dopolnilo teoretične koncepte.

LITERATURA

[1] M. J. Hossain Faruk *idr.*, »A Systematic Literature Review of Decentralized Applications in Web3: Identifying Challenges and Opportunities for Blockchain Developers«, *Proceedings - 2024 IEEE International Conference on Big Data, BigData 2024*, str. 6240–6249, 2024, doi: 10.1109/BIGDATA62323.2024.10826066.

[2] D. Macrinici, C. Cartofeanu, in S. Gao, »Smart contract applications within blockchain technology: A systematic mapping study«, *Telematics and Informatics*, let. 35, št. 8, str. 2337–2354, dec. 2018, doi: 10.1016/J.TELE.2018.10.004.

[3] »Hype Cycle for Web3 and Blockchain, 2024«. Pridobljeno: 15. maj 2025. [Na spletu]. Dostopno na: https://www.gartner.com/en/documents/5623191

[4] N. P. Imperius in A. D. Alahmar, »Systematic Mapping of Testing Smart Contracts for Blockchain Applications«, *IEEE Access*, let. 10, str. 112845–112857, 2022, doi: 10.1109/ACCESS.2022.3216874.

[5] R. Sujeetha in C. A. S. Deiva Preetha, »A Literature Survey on Smart Contract Testing and Analysis for Smart Contract Based Blockchain Application Development«, *Proceedings - 2nd International Conference on Smart Electronics and Communication, ICOSEC 2021*, str. 378–385, 2021, doi: 10.1109/ICOSEC51865.2021.9591750.

[6] L. Marchesi, M. Marchesi, R. Tonelli, in M. I. Lunesu, »A blockchain architecture for industrial applications«, *Blockchain: Research and Applications*, let. 3, št. 4, str. 100088, dec. 2022, doi: 10.1016/J.BCRA.2022.100088.

[7] A. Di Sorbo, S. Laudanna, A. Vacca, C. A. Visaggio, in G. Canfora, »Profiling gas consumption in solidity smart contracts«, *Journal of Systems and Software*, let. 186, str. 111193, apr. 2022, doi: 10.1016/J.JSS.2021.111193.

[8] Z. A. Khan in A. S. Namin, »A Survey of Vulnerability Detection Techniques by Smart Contract Tools«, *IEEE Access*, let. 12, str. 70870–70910, 2024, doi: 10.1109/ACCESS.2024.3401623.

[9] N. Six, N. Herbaut, in C. Salinesi, »Blockchain software patterns for the design of decentralized applications: A systematic literature review«, *Blockchain: Research and Applications*, let. 3, št. 2, str. 100061, jun. 2022, doi: 10.1016/J.BCRA.2022.100061.

[10] N. Sánchez-Gómez, L. Morales-Trujillo, J. J. Gutiérrez, in J. Torres-Valderrama, »The importance of testing in the early stages of smart contract development life cycle«, *Journal of Web Engineering*, let. 19, št. 2, str. 215–242, jun. 2020, doi: 10.13052/JWE1540-9589.1925.

[11] C. Castellon, S. Roy, P. Kreidl, A. Dutta, in L. Boloni, »Energy Efficient Merkle Trees for Blockchains«, *Proceedings - 2021 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom 2021*, str. 1093–1099, 2021, doi: 10.1109/TRUSTCOM53373.2021.00149.

[12] »Popoln vodnik za Ethereum«. Pridobljeno: 11. avgust 2025. [Na spletu]. Dostopno na: https://ethereum.org/sl/

[13] »Web3 Architecture and How It Compares to Traditional Web Apps - The New Stack«. Pridobljeno: 25. marec 2025. [Na spletu]. Dostopno na: https://thenewstack.io/web3-architecture-and-how-it-compares-to-traditional-web-apps/

[14] F. Wessling, C. Ehmke, M. Hesenius, in V. Gruhn, »How much blockchain do you need?: Towards a concept for building hybrid DApp architectures«, v *Proceedings - International Conference on Software Engineering*, IEEE Computer Society, maj 2018, str. 44–47. doi: 10.1145/3194113.3194121.

[15] M. Farnaghi in A. Mansourian, »Blockchain, an enabling technology for transparent and accountable decentralized public participatory GIS«, *Cities*, let. 105, str. 102850, okt. 2020, doi: 10.1016/J.CITIES.2020.102850.

[16] »web3.js - Ethereum JavaScript API — web3.js 1.0.0 documentation«. Pridobljeno: 11. avgust 2025. [Na spletu]. Dostopno na: https://web3js.readthedocs.io/en/v1.10.0/

[17] »Ethers«. Pridobljeno: 11. avgust 2025. [Na spletu]. Dostopno na: https://docs.ethers.org/v5/

[18] H. Chu, P. Zhang, H. Dong, Y. Xiao, S. Ji, in W. Li, »A survey on smart contract vulnerabilities: Data sources, detection and repair«, *Inf Softw Technol*, let. 159, str. 107221, jul. 2023, doi: 10.1016/J.INFSOF.2023.107221.

[19] »GlassFish Wiki : GlassFishVsTomcat«. Pridobljeno: 11. april 2025. [Na spletu]. Dostopno na: https://javaee.github.io/glassfish/wiki-archive/GlassFishVsTomcat.html

[20] Y. Liu, F. R. Yu, X. Li, H. Ji, in V. C. M. Leung, »Blockchain and Machine Learning for Communications and Networking Systems«, *IEEE Communications Surveys and Tutorials*, let. 22, št. 2, str. 1392–1431, apr. 2020, doi: 10.1109/COMST.2020.2975911.

[21] V. Rajasekar, S. Sondhi, S. Saad, in S. Mohammed, »Emerging design patterns for blockchain applications«, *ICSOFT 2020 - Proceedings of the 15th International Conference on Software Technologies*, str. 242–249, 2020, doi: 10.5220/0009892702420249.

[22] X. Xu, C. Pautasso, L. Zhu, Q. Lu, in I. Weber, »A pattern collection for blockchain-based applications«, *ACM International Conference Proceeding Series*, jul. 2018, doi: 10.1145/3282308.3282312.

[23] P. van Vulpen, H. Heijnen, S. Mens, T. Kroon, in S. Jansen, »Upgradeable diamond smart contracts in decentralized autonomous organizations«, *Frontiers in Blockchain*, let. 7, str. 1481914, dec. 2024, doi: 10.3389/FBLOC.2024.1481914/BIBTEX.

[24] »Gas (Ethereum): How Gas Fees Work on the Ethereum Blockchain«. Pridobljeno: 8. avgust 2025. [Na spletu]. Dostopno na: https://www.investopedia.com/terms/g/gas-ethereum.asp

[25] »Understanding Ethereum Gas Fees in 2025: A Comprehensive Guide | KuCoin Learn«. Pridobljeno: 16. maj 2025. [Na spletu]. Dostopno na: https://www.kucoin.com/learn/web3/understanding-ethereum-gas-fees

[26] »Ethereum Average Gas Price Chart | Etherscan«. Pridobljeno: 16. maj 2025. [Na spletu]. Dostopno na: https://etherscan.io/chart/gasprice

[27] A. Vacca, A. Di Sorbo, C. A. Visaggio, in G. Canfora, »A systematic literature review of blockchain and smart contract development: Techniques, tools, and open challenges«, *Journal of Systems and Software*, let. 174, str. 110891, apr. 2021, doi: 10.1016/J.JSS.2020.110891.

[28] N. Sánchez-Gómez, L. Morales-Trujillo, J. J. Gutiérrez, in J. Torres-Valderrama, »The importance of testing in the early stages of smart contract development life cycle«, *Journal of Web Engineering*, let. 19, št. 2, str. 215–242, jun. 2020, doi: 10.13052/JWE1540-9589.1925.

[29] »Foundry - Ethereum Development Framework«. Pridobljeno: 11. avgust 2025. [Na spletu]. Dostopno na: https://getfoundry.sh/forge/tests/overview/

[30] »Hardhat - Ethereum development environment«. Pridobljeno: 11. avgust 2025. [Na spletu]. Dostopno na: https://hardhat.org/hardhat-runner/docs/guides/test-contracts

[31] N. P. Imperius in A. D. Alahmar, »Systematic Mapping of Testing Smart Contracts for Blockchain Applications«, *IEEE Access*, let. 10, str. 112845–112857, 2022, doi: 10.1109/ACCESS.2022.3216874.

[32] A. Reyes, M. Jimeno, in R. Villanueva-Polanco, »Continuous and Secure Integration Framework for Smart Contracts«, *Sensors 2023, Vol. 23, Page 541*, let. 23, št. 1, str. 541, jan. 2023, doi: 10.3390/S23010541.

[33] T. Chen *idr.*, »GasChecker: Scalable Analysis for Discovering Gas-Inefficient Smart Contracts«, *IEEE Trans Emerg Top Comput*, let. 9, št. 3, str. 1433–1448, 2021, doi: 10.1109/TETC.2020.2979019.

[34] T. Chen, X. Li, X. Luo, in X. Zhang, »Under-Optimized Smart Contracts Devour Your Money«, *SANER 2017 - 24th IEEE International Conference on Software Analysis, Evolution, and Reengineering*, str. 442–446, mar. 2017, doi: 10.1109/SANER.2017.7884650.

[35] L. Palechor in C. P. Bezemer, »How are Solidity smart contracts tested in open source projects? An exploratory study«, *Proceedings - 3rd ACM/IEEE International Conference on Automation of Software Test, AST 2022*, str. 165–169, 2022, doi: 10.1145/3524481.3527228.

[36] Y. Huang, Y. Bian, R. Li, J. L. Zhao, in P. Shi, »Smart contract security: A software lifecycle perspective«, *IEEE Access*, let. 7, str. 150184–150202, 2019, doi: 10.1109/ACCESS.2019.2946988.

Mitja Gradišnik je asistent na Fakulteti za elektrotehniko, računalništvo in informatiko Univerze v Mariboru. Njegovo raziskovalno delo je usmerjeno v sodobne pristope k razvoju programskih rešitev, zagotavljanje kakovosti programskih produktov ter uporabo metod podatkovnega rudarjenja v programskem inženirstvu. Sodeluje pri številnih raziskovalnih in aplikativnih projektih, ki potekajo v okviru Inštituta za informatiko.

▪

Tina Beranič je docentka na Fakulteti za elektrotehniko, računalništvo in informatiko Univerze v Mariboru. Doktorirala je leta 2018 iz tematike identifikacije pomanjkljive programske kode. Njeno raziskovalno delo obsega domeno kakovosti programske opreme, še posebej področje programskih metrik in mejnih vrednosti ter njihove uporabe za namen vrednotenja programske opreme. Ukvarja se tudi s področjem revizije informacijskih sistemov, pri čemer je leta 2017 pridobila certifikat CISA (Certified Information Systems Auditor) in leta 2020 naziv Preizkušena revizorka informacijskih sistemov na Slovenskem inštitutu za revizijo.

▪

Muhamed Turkanović je visokošolski učitelj, izredni profesor, na Fakulteti za elektrotehniko, računalništvo in informatiko Univerze v Mariboru. Je vodja raziskovalne skupine Blockchain Lab:UM Inštituta za informatiko, namestnik predstojnika Inštituta za informatiko, vodja slovenskega EDIH-a DIGI-SI, vodja projektov H2020, Horizont Evropa, Interreg Alpine Space ter ARRS CRP. Njegovi trenutni raziskovalni interesi vključujejo področja tehnologij veriženja blokov, podatkovnih tehnologij ter digitalnih identitet.

Iz Islovarja

Islovar je spletni terminološki slovar informatike, ki ga objavlja jezikovna sekcija Slovenskega društva INFORMATIKA na naslovu http://www.islovar.org. Slovar je javno dostopen za vpoglede in vnašanje novih izrazov.

aktívna napráva -e -e ž (angl. active device) naprava, ki za delovanje potrebuje poseben vir energije; prim. pasivna naprava

kazálna napráva -e -e ž (angl. tracking device) naprava za upravljanje premikanja kazalca na računalniškem zaslonu

naménska napráva -e -e ž (angl. appliance) naprava, namenjena za določena opravila

navíti -ijem dov. (angl. overclock) s tehničnimi prilagoditvami povečati takt naprave nad tistega, za katerega je bil projektiran

pasívna napráva -e -e ž (angl. passive device) naprava, ki za delovanje ne potrebuje posebnega vira energije; prim. aktivna naprava

prostórska merílna napráva -e -e -e ž (angl. coordinate measuring machine, CMM) naprava za merjenje geometrijskih značilnosti predmeta

róbna napráva -e -e ž (angl. edge device) naprava, ki omogoča dostop do omrežja; prim. jedrno omrežje

sledílna napráva -e -e ž (angl. tracking device) naprava za določitev ali sledenje položaja ali gibanja na daljavo

TROIA

OPTIMIZIRAMO PRIHODNOST VAŠEGA PODJETJA

IZKUŠENA EKIPA

Nudimo sodelovanje z izkušeno ekipo strokovnjakov, ki je predana zagotavljanju individualiziranih rešitev za vsako stranko.

PRILAGOJENE REŠITVE

Ponujamo rešitve, ki so prilagojene potrebam vsake stranke.

PREVERJENI REZULTATI

Imamo dokazano uspešnost zaključenih projektov in zadovoljnih strank v različnih panogah in na različnih področjih.

REŠITVE ZA VAS

- **Rešitve za vzdrževanje in upravljanje premoženja podjetja**
- **AR rešitve za vizualizacijo GIS podatkov na terenu**
- **Upravljanje IT sredstev**
- **Upravljanje velepodatkov**

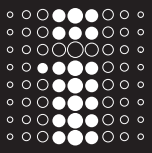
www.troia.eu info@troia.si

Izpitni centri ECDL

ECDL (European Computer Driving License), ki ga v Sloveniji imenujemo evropsko računalniško spričevalo, je standardni program usposabljanja uporabnikov, ki da zaposlenim potrebno znanje za delo s standardnimi računalniškimi programi na informatiziranem delovnem mestu, delodajalcem pa pomeni dokazilo o usposobljenosti. V Evropi je za uvajanje, usposabljanje in nadzor izvajanja ECDL pooblaščena ustanova ECDL Fundation, v Sloveniji pa je kot član CEPIS (Council of European Professional Informatics) to pravico pridobilo Slovensko društvo INFORMATIKA. V državah Evropske unije so pri uvajanju ECDL močno angažirane srednje in visoke šole, aktivni pa so tudi različni vladni resorji. Posebno pomembno je, da velja spričevalo v 148 državah, ki so vključene v program ECDL. Doslej je bilo v svetu v program certificiranja ECDL vključenih že preko 16 milijonov oseb, ki so uspešno opravile preko 80 milijonov izpitov in pridobile ustrezne certificate. V Sloveniji je bilo doslej v program certificiranja ECDL vključenih več kot 18.000 oseb in opravljenih več kot 92.000 izpitov. V Sloveniji sta akreditirana dva izpitna centra ECDL, ki imata izpostave po vsej državi.

AGORA

Micro Team



Znanstveni prispevki

Daniel Kovačevič Rudolf, Ana Malešič
ALTERNATIVE ČEZMEJNIM SPLETNIM PLAČILNIM STORITVAM, RAZVITE S PRISTOPOM ŽIVIH LABORATORIJEV

Uroš Godnov
DELOVANJE ALGORITMA JARO-WINKLER GLEDE NA MESTO POJAVLJANJA TIPOGRAFSKIH NAPAK

Strokovni prispevki

Aleš Gros
POGLED NA DANAŠNJE IN PRIHODNJE IZZIVE INFORMATIKE V ZDRAVSTVU: OD POVEZLJIVOSTI DO ANALITIČNE POMOČI PRI DIAGNOSTICIRANJU IN ZDRAVLJENJU

Olga Šušteršič, Uroš Rajkovič
INFORMACIJSKA PODPORA ODLOČANJU V PROCESU ZDRAVSTVENE NEGE

Rok Bojanc, Boris Šušmak
LOGICAL - PLATFORME RAČUNALNIŠTVA V OBLAKU IN ORODJA ZA LOGISTIČNE CENTRE IN SKUPNOSTI

Informacije

IZ ISLOVARJA

KOLEDAR PRIREDITEV

ISSN 1318-1882