Analiza nefunkcionalnih zahtev na primeru uporabe priporočilnega sistema pametnih pogodb

Sandi Gec, Vlado Stankovski
Univerza v Ljubljani, Fakulteta za računalništvo in informatiko, Večna pot 113, 1000 Ljubljana
sandi.gec@fri.uni-lj.si, vlado.stankovski@fri.uni-lj.si

Izvleček

Nefunkcionalne zahteve so opredeljena kot ena izmed ključnih meril kakovosti programske opreme, ki vplivajo na zmogljivost, uporabnost, varnost in zanesljivost. Tehnologija veriženja blokov razširja nabor običajnih nefunkcionalnih zahtev z dodatnimi atributi, kot so preglednost, nespremenljivost in decentralizacija. Integracija tehnologije veriženja blokov običajno osredotoča na funkcionalne zahteve, pri čemer nefunkcionalne pogosto niso dovolj naslovljene. Zato se nefunkcionalne zahteve običajno analizirajo z namenskimi vmesniki uporabniškega programa, spletnimi pajki in drugimi orodji, ki zahtevajo nenehno vzdrževanje ter prilagajanje delovanja tovrstnih orodij. V tem delu predlagamo analizo nefunkcionalnih zahtev za rešitve tehnologije veriženja blokov, ki temelji na umetni inteligenci, kjer so uporabljena orodja, kot so Copilot, ChatGPT in druga. Takšna orodja uporabljamo za identifikacijo, vrednotenje in optimizacijo nefunkcionalnih zahtev za rešitve verig blokov na bolj celovit način. Rezultati naše raziskave so predstavljeni na primeru uporabe priporočljivega sistema, ki vključuje diskusijo o tem, kako izboljšati dolgoročno vzdržnost predlagane rešitve.

Ključne besede: umetna inteligenca, trilema, Ethereum virtualni stroj, nefunkcionalne zahteve

Analysis of non-functional requirements for smart contract recommender system

Abstract

Non-functional requirements (NFRs) are essential criteria of software quality that affect the performance, usability, security, and reliability of software. The blockchain, as a novel technology expands the set of common NFRs with additional attributes, such as transparency, immutability, and decentralization. However, blockchain integration usually focuses on functional requirements (FRs), while NFRs are not sufficiently addressed. Therefore, NFRs are usually analysed with dedicated APIs, crawlers, and other tools that require ongoing maintenance and manual intervention. This paper proposes an AI-based analysis of NFRs for blockchain solutions, where AI tools such as Copilot, ChatGPT, and others are leveraged. We use these tools to generate, evaluate, and optimize NFRs for blockchain solutions in a more comprehensive way. The results of our solution are presented on a recommender system use case, including explanation on how to improve the sustainability of the proposed solution.

Keywords: Artificial Inteligence, trilemma, Ethereum Virtual Machine, Non-Functional Requirements

1 UVOD

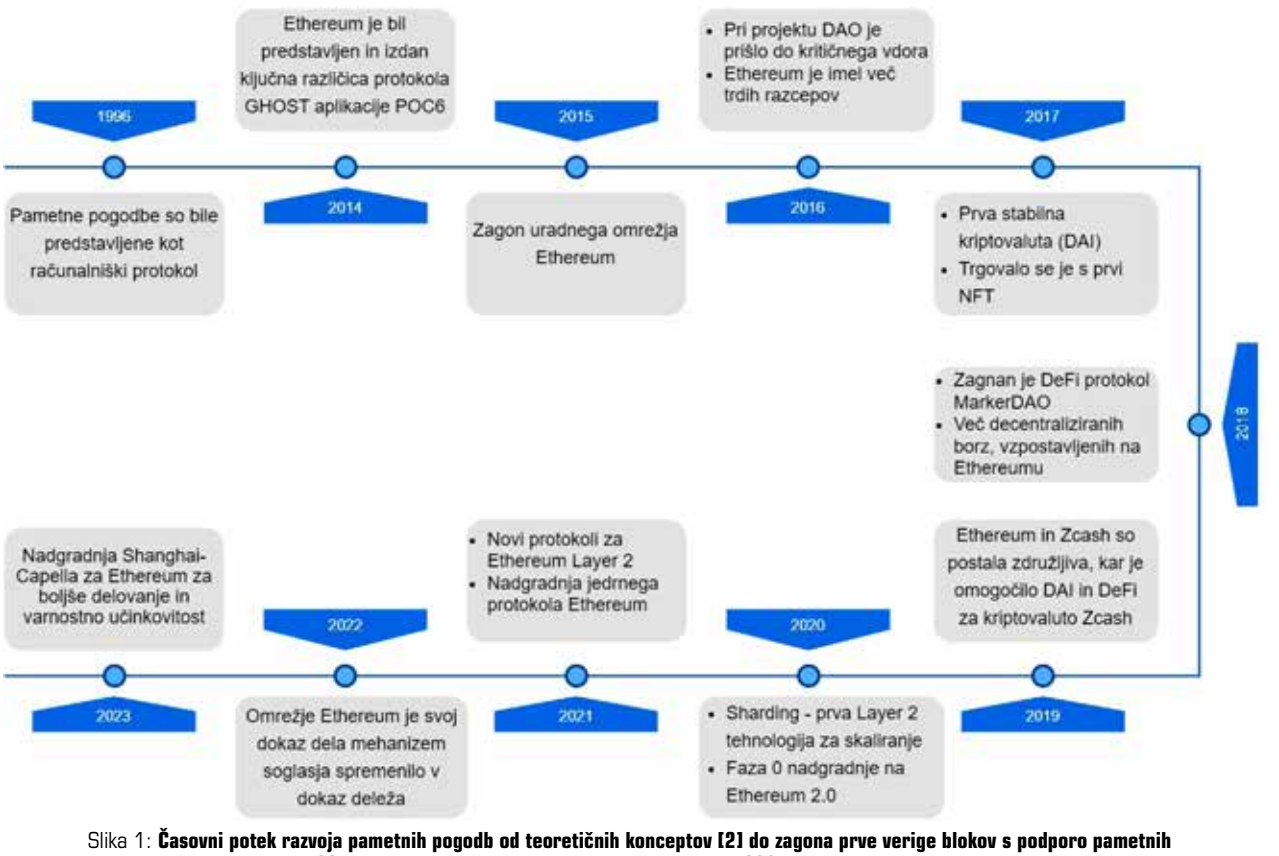
Na področju programskega inženiringa so nefunkcionalne zahteve ključnega pomena, še posebej pri oblikovanju programske opreme, analizi programske opreme in splošnem razumevanju programske opreme. Ker programska oprema običajno ni monolitna, centralizirana ali homogena, temveč heterogena, sestavljena iz več komponent in v nekaterih primerih decentralizirana (npr. decentralizirane baze podatkov), je pozornost na nefunkcionalnih zahtevah bistvena. To velja še posebej na področju tehnologije

veriženja blokov, kjer je kompleksnost programske opreme pogosto zelo visoka.

Pri razvijajočih se tehnologijah distribuiranih razpršenih knjig (angl. Distributed Ledger Technology, DLT) je pomembno temeljito razumevanje nefunkcionalnih zahtev za izboljšanje procesa izbire razpršene knjige. Naša osredotočenost je na DLT, ki temelji na Ethereumu, in vključujejo podporo za pametne pogodbe

Solidity prek komponente Ethereum virtualnega stroja (angl. Ethereum Virtual Machine, EVM), ki deluje v primerih vozlišč razpršene knjige. Čeprav primarna razpršena knjiga Ethereum mainnet morda ni primerna za vse primere uporabe, ki zahtevajo določene nefunkcionalne zahteve, je ključna za razvoj funkcionalnosti pametnih pogodb, saj se knjiga nenehno ter skrbno razvija [1], kot prikazuje Slika 1.

Zato postaja preučevanje razpoložljive razpršene knjige, ki omogočajo EVM, z vidika nefunkcionalnih zahtev. Čeprav se funkcionalne zahteve med različnimi platformami pogosto ujemajo, je razlika v nefunkcionalnih zahtevah znatna. V tej raziskavi bomo predstavili celovito analizo razpršenih knjig, ki temeljijo na EVM. Najprej se bomo lotili najbolj standardne analize nefunkcionalnih zahtev, imenovane trojček razpršene knjige, in preučili značilnosti ter razlike med vsemi razpoložljivimi plastmi EVM, vključno s Plastjo-0, ki ni EVM. Ker ni standardov za opredelitev zahtev pri EVM, bomo raziskali najbolj izstopajoče nefunkcionalne zahteve z uporabo storitev umetne inteligence. Razčlenili bomo razlike med storitvami umetne inteligence in analizirali rezultate umetne inteligence, kot so predlagane nefunkcionalne zahteve. Nadalje bomo predstavili integracijo nefunkcionalnih zahtev v primeru uporabe priporočilnega sistema za pametne pogodbe in opisali delovni tok nefunkcionalnih zahtev ter morebitne koristi v okviru primera uporabe. V eksperimentalni študiji bomo analizirali storitve umetne inteligence v kontekstu nefunkcionalnih zahtev, ki omogočajo EVM. Razčlenili bomo najbolj prevladujoče nefunkcionalne zahteve v kontekstu predstavljenega primera uporabe.



Slika 1: **Časovni potek razvoja pametnih pogodb od teoretičnih konceptov [2] do zagona prve verige blokov s podporo pametnih pogodb Ethereum v letu 2015 ter vse nadaljnje pomembne mejnike. V letu 2024 je odmevna nadgradnja Ethereum Dencun, ki zmanjša stroške transakcij na verigah Plast-2.**

Preostanek članka bo sledil naslednjim korakom. Razdelek 2 predstavi aktualne raziskovalne dosežke in identificira vrzel, ki jo obravnava ta študija. Razdelek 3 predstavi osnovne koncepte študije, vključno z Ethereum plasti in storitvami umetne inteligence. Razdelek 4 pojasni integracijo nefunkcionalnih zahtev v motivacijskem primeru uporabe. Razdelek 5 predstavi eksperimentalno študijo. Razdelek 6 razpravlja o izvedenih eksperimentih in podaja zaključke.

2 SORODNA DELA

Znanstvene raziskave nefunkcionalnih zahtev programskih rešitev so aktualne že desetletja, pri čemer se je v zgodnjih raziskavah definirala osnovna taksonomija z izhodiščnimi kategorijami, kot so: (i) funkcionalnosti obnašanja, (ii) performančne lastnosti (npr. časovne, prostorske, hitrostne in druge), (iii) kvalitativne lastnosti (npr. uporabnost, varnost, zanesljivost in druge) in (iv) druge lastnosti (npr. pravni vidik, okoljski in drugi) [3]. Temeljne ugotovitve raziskave leta 2007 so obsegale programske rešitve predvsem monolitnih ter centraliziranih aplikacij in niso vključevale računalništva v oblaku, ki je bil v takrat v povojih. Pomemben primer decentralizirane aplikacije (angl. decentralised Applications, dApps) je bil predstavljen leta 2009 z zagonom prve javne rešitve, ki temelji na tehnologiji veriženja blokov s strani neznanega avtorja s psevdonimom Satoshi Nakamoto [4]. Zanimanje za tehnologijo veriženja blokov je privedlo do raziskav ter razvoja, ki je privedla do izdaje javnih kriptovalut z različnimi nameni. Določene verige blokov so bile predlagane z namenom izboljšav verige blokov Bitcoin, druge so se pa osredotočale na nefunkcionalne zahteve, kot je zasebnost v korelaciji z anonimnostjo ter različnih raziskovalnih vej [5]. Precej raziskovalnega zanimanja na področju tehnologije veriženja blokov je bilo naklonjenega primeru uporabe samostojna suverena identiteta (angl. Self-Sovereign Identity, SSI) [6], ki je še dandanes izjemno aktualna tudi s širše uporabe (npr. na nivoju Evropske unije). Pri pregledu aktualnih raziskovalnih publikacij so avtorji identificirali ter opredelili 22 nefunkcionalnih zahtev aktualnih na področju SSI.

Zanimiv koncept je bil predstavljen z zagonom verige blokov Ethereum [7], kjer so avtorji predstavili možnost decentraliziranih skript imenovane pametne pogodbe. Z vidika nefunkcionalnih zahtev so M. Staderini idr. [8] predstavili problematiko izbire ustrezne verige blokov, kjer so predlagali metodologijo izbire na podlagi namenskih diagramov odločanja. Posebno pozornost so avtorji namenili identifikaciji nefunkcionalnih zahtev, pri čemer so se osredotočili na njihovo stopnjo pomembnosti v različnih tipih verig blokov: javne, zasebne, stopnje dovoljenj uporabnikov in odločanje po konceptu konzorcija. Raziskovalne ugotovitve avtorjev nudijo izhodišče za nadaljnjo analizo nefunkcionalnih zahtev, ki so obravnavane v našem delu na javnih verigah blokov. Analizo nefunkcionalnih zahtev praktičnih produkcijskih primerov uporabe podprte s tehnologijo veriženja blokov je predstavil M. Kassab [9] in sicer tako, da je najprej opredelil identificirane nefunkcionalne zahteve ter nato ocenil njihovo pomembnost glede na krovne funkcionalnosti v primerih uporabe. Na podlagi 1327 zahtev zastopanih v 7 različnih produkcijskih primerih uporabe je avtor identificiral ter opredelil 6 osnovnih razredov nefunkcionalnih zahtev: (i) zasebnost, (ii) skalabilnost/performančnost, (iii) interoperabilnost, (iv) uporabnost, (v) skladnost z zakonodajo (npr. regulatorji) in (vi) Operativne in finančne omejitve (npr. zadostno število vozlišč, velikost blokov idr.). V našem delu se poleg omenjenih zahtev osredotočamo predvsem na tiste, ki se pogosto pojavljajo kot implicitne (npr. v arhitekturni opredelitvi plasti EVM verig blokov) in eksplicitne (npr. kvantitativne) lastnosti.

Pregled najpomembnejših pristopov inženiringa programskih zahtev na področju tehnologij veriženja blokov so bili predstavljeni s strani M. S. Farooq idr. [10] pri čemer je bil pregled del med leti 2015 in 2021 kritično analiziran ter izpostavljene omejitve posameznih pristopov na izbranih domenah. Pomembno delo predstavlja tudi pregledni znanstveni prispevek, ki izpostavlja raziskovalne izzive, kot so integracija zahtev, identifikacija metrik in predstavitev zahtev v sistemih s podporo tehnologije veriženja blokov [11]. V primerjavi z našim delom, kjer sta najbolj zastopani domeni oblačnega računalništva ter tehnologije veriženja blokov s podporo pametnim pogodbam, v predhodnih raziskavah tovrstna domena ni bila zadostno obravnavana.

V našem delu se osredotočamo na verige blokov, ki omogočajo EVM in posledično omogočajo izvajanje Solidity pametnih pogodb. Zanimiv pristop z uporabo odločitvenih modelov za najpogostejše funkcionalnosti (npr. avtentikacija, avtorizacija, komunikacija izven verige idr.) s katerimi so pri modeliranju modelov upoštevane tudi nefunkcionalne

zahteve. Čeprav je največja omejitev predlaganih modelov rigidnost (npr. monetizacijski modeli so le minimalen del kapacitet, ki jih omogočajo pametne pogodbe), le-ti predstavljajo dobro izhodišče za razumevanje zahtev v vsakdanjih primerih uporabe. Podobno je predstavljena možnost izbire verige blokov v odvisnosti od različnih vidikov kot je možnost migracije na druge verige blokov ali najpogosteje funkcionalnosti, ki jih veriga blokov omogoča [12]. Avtorji so na teoretičnem modelu igre predlagali matematično analizo s katere so ugotovili, da uporabniki EVM verig blokov niso osredotočeni zgolj na natanko eno verigo blokov. Izbira ustrezne EVM verige blokov glede na primer uporabe pametnih pogodb je ključnega pomena, da lahko zagotavljamo pričakovano kakovost storitve ter kakovost izkušnje.

Omenjeni pristopi so nas motivirali, da smo v našem preteklem delu predlagali analizo EVM verig blokov za primere uporabe internet stvari z nefunkcionalnega vidika, bolj natančno smo se osredotočili na hitrost transakcije in ceno transakcij [13]. V analizi smo s pomočjo vmesnikov uporabniškega programa (angl. Application Programming Interface, API) in s pomočjo namenskih spletnih pajkov ter ostalih orodij predlagali primerjalno analizo med obetavnimi EVM verigami blokov.

3 NEFUNKCIONALNE ZAHTEVE ZA VERIGE BLOKOV S PODPORO EVM

V tem poglavju je najprej opredeljen pomen večplastnega razvoja, ki temelji na težavah osnovnega Ethereum omrežja. Omrežje je kot prvo izjemno decentralizirano ter posledično varno, po drugi strani pa so transakcije počasnejše ter drage, kar v splošnem povzamemo kot manj skalabilno. Za boljše razumevanje nefunkcionalnih zahtev primerjamo različne storitve umetne inteligence, ki nam praviloma bolj podrobno opredelijo nefunkcionalne zahteve verig blokov.

3.1 Analiza plasti

Z razvojem tehnologije veriženja blokov, ki je bila prvotno pretežno decentralizirana, so se pojavile težave pri možnosti podpore različnim primerom uporabe. Pri višji stopnji decentralizacije je posledično skalabilnost nižja. Obenem so aplikacije z nižjo stopnjo decentralizacije manj varne zaradi večje izpostavljenosti vozlišč ter možnosti raznih napadov (npr. 51% napad). Problematiko kompromisa omenjenih nefunkcionalnih zahtev (varnost, skalabilnost in decentralizacija) je opredelil ustanovitelj Ethereum verige blokov Vitalik, in sicer kot trilemo (angl. trilemma) tehnologije veriženja blokov. Glede na različne namene uporabe Ethereum verige blokov ter podpor različnih primerov uporabe, so se razvijalci odločili razvit več plasten pristop, kjer so trilema lastnosti različno zastopani. Iz prvotne plasti Ethereum omrežja (Plast-1) se je razvilo skupno pet plasti, kjer so glavne lastnosti ponazorjene v Tabeli 1.

3.2 Pristopi umetne inteligence za analizo nefunkcionalnih zahtev

Javne storitve umetne inteligence so postale pomembna orodja, ki nam omogočajo učenje na podlagi interakcije v obliki klepeta s sistemom. Na

Tabela 1: **Osnovne plasti, ki opredeljujejo EVM verige blokov.**

Ime plasti	Varnostni mehanizem	Odvisnost	Stopnja skalabilnosti	Lastnosti	Primeri verig
Plast-0	Lastni (deli paraverigam)	Neodvisen	Omejena	Deluje kot zanašajoča se veriga, s poudarkom na varnosti in interoperabilnosti med glavnimi paraverigami	Polkadot [14]
Plast-1	Lastni	Neodvisen	Omejena	Zajema podatkovne, omrežne, konsenzne in aktivacijske podplasti v logični arhitekturi	Bitcoin, Ethereum
Plast-2	Se zanašajo na Plast-1	Zgrajen na temeljih Plasti-1	Višja	Neodvisen ali vzporedno, si prizadevajo za reševanje izzivov transakcijske hitrosti in razširljivosti	Polygon, Optimism
Plast-3	Se zanašajo na Plast-2	Zgrajen na temeljih Plasti-2	Višja	Aplikacijska plast, gosti dApps, interoperabilna, razdeljena na dve podplasti: aplikacijska in izvajalna	Uniswap, Axie Infinity
Stranska veriga (angl. Sidechain)	Lastni	Neodvisen	Višja	Povezuje z nadrejeno verigo preko dvosmernega mostu	Cardano Milkomeda V2

tak način sistem bolje razume kontekst iskanja kot običajni iskalniki, čeprav marsikateri brskalniki beležijo zgodovino našega iskanja in nam, zato lahko vračajo bolj relevantne rezultate. Storitve umetne inteligence je mogoče uporabljati v vsakdanjem življenju, kot orodje za izobraževanje, reševanje nalog in drugo [15]. V Tabeli 2 je predstavljen nabor primerov aktualnih javnih storitev umetne inteligence, pri čemer so, med drugimi, opredeljene tudi lastnosti, ki so pomembne v kontekstu integracije. Pri pedagoškem področju postajajo takšna orodja problematična, saj so storitve zmožne reševanja marsikaterega problema, tudi programerskih. Kljub temu je treba rezultate kritično preveriti, saj se mnogokrat zgodi, da rešitev v obliki programske kode ne prevede, narobe deluje ali nam podaja napačen rezultat. V našem delu uporabljamo storitve umetne inteligence kot del priporočilnega sistema, predvsem kot dopolnilo obstoječih informacij, predvsem kvantitativnih, o EVM verigah blokov. V našem delu se osredotočamo zgolj na brezplačne različice storitev umetne inteligence.

4 INTEGRACIJA ZAHTEV V OBLAČNE REŠITVE

V tem poglavju najprej predstavimo osnovne značilnosti priporočilnega sistema pametnih pogodb. Nato predstavimo potek integracije nefunkcionalnih zahtev kot del sistema.

4.1 Primer uporabe priporočilnega sistema pametnih pogodb

Priljubljenost oblačnih ter sorodnih arhitektur, kot so v megli (angl. Fog) in na robu (angl. Edge) je v nenehnem vzponu zaradi možnosti dodeljevanja namenskih nalog na različne nivoje ter možnost integracije različnih protokolov komunikacije. V tovrstnih arhitekturah so lahko vozlišča del podatkovnih centrov, pri čemer del infrastrukture lahko zagotavljajo končni uporabniki [16]. Postopek razbremenitve vključuje končne uporabnike, ki zagotavljajo storitve ali vire v infrastrukturi megle, kar običajno zahteva ročne operacije, kot so dogovor o ceni in politiki, registracija nove komponente zagotavljanja in drugo. Te ročne operacije se pogosto izvajajo prek centraliziranih komponent. Za poenostavitev takšnih procesov je mogoče ročne operacije opredeliti v namenskih pametnih pogodbah, ki upoštevajo sistemske zahteve in potrebe deležni- kov. Razvoj tako zapletenih pametnih pogodb je izziv in zahteva izkušenega razvijalca pametnih pogodb. Zato smo predlagali priporočilni sistem, ki kot rezultat vrača predloge pametnih pogodb, ki predstavljajo preverjeno varno programsko kodo ter so namenjene razširitvam za podporo primerom uporabe. Prednost takšnega priporočilnega sistema je povzeta kot izboljšanje procesa integracije pametnih pogodb s predlogi pametnih pogodb, ki se vračajo na podlagi oblačne arhitekture sistema ter funkcionalne zahtev, ki jih izbere razvijalec [17].

Tabela 2: **Primeri aktualnih javnih storitev umetne inteligence**

Ime storitve	Kategorija	Primarni namen	API podpora	Možnosti integracije
ChatGPT	Klepetalnik z umetno inteligenco	Pogovor in pomoč uporabnikom	Da	Da
Microsoft Copilot	Klepetalnik z umetno inteligenco, asistent izvorne kode	Pomoč uporabnikom in razvijalcem programske opreme	Da	Da
Google Bard	Klepetalnik z umetno inteligenco	Pogovor in pomoč uporabnikom	Da	Da
Craiyon AI	AI generator slik	Ustvarjanje slik	Da	Da
HeyWire.ai	B2B SaaS	Novinarstvo in ustvarjanje korporativnih vsebin	Ne	Ne
Eightify	Izločanje glavnih točk iz dolgih videoposnetkov	Razširitev brskalnika	Ne	Ne
Feathery	Pomoč razvijalcem programske opreme	Izdelava prilagojenih obrazcev in delovnih tokov brez kodiranja	Da	Da
Azure AI Vision	Umetna inteligenca kot storitev (angl. Artificial Inteligence as a Service, AIaaS)	Računalniški vid	Da	Da
Cloud Vision API	AIaaS	Računalniški vid	Da	Da
Amazon Rekognition Image	AIaaS	Prepoznavanje slik in globoko učenje	Da	Da

Osnovni scenarij uporabe priporočilnega sistema obravnava pogoste izzive, s katerimi se soočajo mlajši razvijalci, vključno s pomanjkanjem izkušenj pri razvoju pametnih pogodb in omejenim znanjem o potencialnih predlogah pametnih pogodb za večkratno uporabo, oblačnih arhitektur ali oboje. Kot odgovor predstavljenim izzivom smo razvili nov priporočilni sistem pametnih pogodb, zasnovan za doseganje naslednjih ciljev:

- ustvarjanje predlog pametnih pogodb, usklajenih s specifikacijami aplikacije (funkcionalne zahteve), ob upoštevanju najboljših praks in vzorcev oblikovanja;
- podpreti implementacijo verige blokov tako, da razvijalcem interneta stvari ponudi zanesljive, stan- dardizirane pametne pogodbe, prilagojene njihovim primerom uporabe;
- pospešiti splošni razvoj in izvajanje pametnih pogodb v oblačnih okoljih.

Pomemben prispevek takšnega sistema se ne odraža zgolj s splošnega vidika programskega inženiringa za namene razvoja programske opreme, ampak tudi kot pedagoško orodje za izboljšanje razumevanja možnosti razvoja pametnih pogodb. To poteka skozi izbiro različnih funkcionalnih zahtev na osnovni oblačne tri-nivojske arhitekture spletne aplikacije, kjer študentom poenostavi razvoj pametnih pogodb na projektnem delu razvoja spletne aplikacije [18]. Prvotna različica priporočilnega sistema je zasnovana, da vključuje kot vhodne podatke le funkcionalne tipe zahtev. V nadaljevanju je orisan predlog integracije nefunkcionalnih zahtev z namenom olajšanja izbiro ustrezne EVM verige blokov.

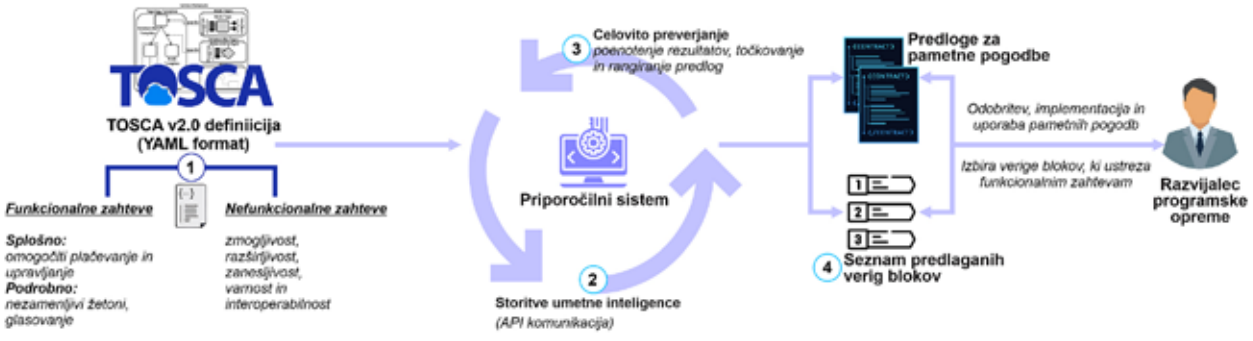
4.2 Integracija nefunkcionalnih zahtev

V osnovni različici priporočilnega sistema so bile naslovljene funkcionalne zahteve, saj vloga nefunkcionalnih zahtev ne bi imela dodane vrednosti pri osnovnem delovanju in algoritmu sistema pri priporočilu pametnih pogodb. Kljub temu je vključitev nefunkcionalnih zahtev v takšen sistem smotrna, saj uspešna integracija pripomore k sledečim izboljšavam:

- boljše razumevanje raznolikosti EVM verig blokov,
- podatki EVM verig blokov v kontekstu zahtev so pridobljeni iz javnih storitev umetne inteligence,
- nov tip izhodnega podatka sistema, kjer so predlagane EVM verige blokov (izhod sistema), ki ustrezajo podanim nefunkcionalnim zahtevam (vhodni podatki) in
- celovit pristop razvoja in integracije pametnih pogodb.

Pri integraciji nefunkcionalnih zahtev osnovni algoritem ter obstoječi vhodni podatki sistema ostanejo nespremenjeni. Zahteve med sabo niso komplementarne, saj vračajo dva različna vhoda. Na Sliki 2 je orisan potek delovanja priporočilnega sistema z integracijo nefunkcionalnih zahtev, kjer so glavni gradniki sistema opisani v naslednjih korakih:

1. Nefunkcionalne zahteve so izbrane s strani razvijalca programske opreme, kjer razvijalec izbere nefunkcionalno zahtevo ter vrednost, ki je lahko:
   - kvantitativna vrednost (npr. povprečna cena transakcije, hitrost potrjevanja transakcij, vključno z interakcijami pametnih pogodb, hitrost generiranja bloka idr.),
   - kvalitativna vrednost, izbrana na podlagi mehko opredeljene lastnosti (npr. skalabilnost, varno- stni mehanizmi idr.) in
   - želja po dodatnih informacijah pri zahtevah, kjer je definicija metrike bodisi preveč kompleksna bodisi se zahteva lahko nanaša na različne lastnosti sistema (npr. varnost).



Slika 2: **Nadgrajena arhitektura priporočilnega sistema pametnih pogodb na podlagi nefunkcionalnih zahtev kot vhodnega podatka sistema vrne seznam predlaganih verig blokov.**

2. Konvencionalni pristopi iskanja kvantitativnih nefunkcionalnih zahtev iz EVM verig blokov [13]. Pristopi lahko vključujejo REST API storitve, beleženje dejavnosti v realnem času na verigi blokov, spletne pajke ter druge vire. Takšen pristop zahteva vzdrževanje logike za pridobivanje podatkov (npr. protokoli, spremembe specifikacij, posodabljanje URL naslovi spletnih virov itd.). Poleg tega se nove EMV verige blokov na novo zaganjajo za pokrivanje namenskih primerov uporabe ter obstoječe EVM verige blokov se nadgrajujejo z novimi koncepti, specifikami in drugimi lastnostmi. Z uporabo storitev umetne inteligence, predvsem klepetalniki (angl. chatbot), je mogoče vzdrževanje minimizirati. V tem koraku je smotrno podatke hraniti v relacijski podatkovni bazi.
3. V koraku izvajanja procesiranja priporočilnega sistema se neodvisno izvajata algoritma funkcionalnih in nefunkcionalnih zahtev. Pri procesiranju nefunkcionalnih zahtev je algoritem opredeljen kot večkriterijsko odločanje (glej Algoritem 1) pri čemer je vsaka zahteva predstavlja kriterij, ki mora biti izpolnjen. Kriteriji se pregledujejo neodvisno na podlagi podatkov iz prejšnje točke in sicer pred procesiranimi podatki pridobljenimi s storitvami umetne inteligence in ostalimi viri s prejšnje točke. V tem koraku je pomembno, da se ponovno preverijo storitve umetne inteligence, saj se modeli podatkov za učenje pogosto posodabljajo. Poleg tega je smotrno uporabiti več virov storitve umetne inteligence komplementarno, saj uporabljajo različne modele, metodologije iskanja podatkov ter metode strojnega učenja, posledično različno podrobne odgovore. Na primer, pri podanem vprašanju *„Ali je bila v zadnjem* času *izvedena kakšna pomembna nadgradnja Ethereum omrežja?"* so odgovori različni. Pri uporabi brezplačne storitve *ChatGPT* pridobimo odgovor o nadgradnjah v preteklem letu 2023. Po drugi strani nam pa storitev *Microsoft Copilot* vrne odgovor o pomembni nadgradnji v letu 2024 in sicer *Dencun*, vključno s ključnimi lastnostmi nadgradnje (glej opis Slike 1).
4. Pri pripravi podatkov je pomembno, da uporabniku podamo seznam verig blokov, ki ustrezajo podanim nefunkcionalnim zahtevam. V primeru, da ni mogoče najti verige, ki bi ustrezala vsem nefunkcionalnim zahtevam, se uporabniku izpiše seznam kandidatov ter poda tudi seznam nefunkci- onalnih zahtev, ki ne ustrezajo zahtevam vhodnih podatkov. Na tak način ima razvijalec programske opreme možnost vpogleda v pogojno sprejemljive rezultate.

5 EVALVACIJA

V tem poglavju predstavimo eksperimentalno študijo simulacije priporočilnega sistema EVM verig blokov na podlagi uporabnikovih nefunkcionalnih zahtev. Študija obsega simulacijo treh pogostih primerov uporabe, in sicer:

- Internet stvari (angl. Internet of Things, IoT),
- decentralizirane finance (angl. Decentralized Finance, DeFi) in
- lastništvo na podlagi nezamenljivih žetonov (angl. Non-fungible tokens, NFT).

Najprej določimo relevantne nefunkcionalne zahteve, ki smo jih pridobili s povpraševanjem v brezplačnih različic orodij *ChatGPT* in *Microsoft Copilot*. Skupni predlogi relevantnih nefunkcionalnih zahtev so: (i) varnost, (ii) skalabilnost (npr. odzivni čas oz. hitrost transakcije), (iii) zmogljivost ter učinkovitost (npr. hitrost transakcij), (iv) zanesljivost ter dostopnost, (v) mehanizem soglasja, (vi) celovitost ter doslednost podatkov ali integriteta, in (vii) interoperabilnost.

Izmed nabora identificiranih zahtev se bomo osredotočili zgolj na takšne, kjer je mogoče določiti kvantitativno ali kvalitativno vrednost. Na zahteve, ki so zgolj informativne narave ali jih je v osnovi težje interpretirati (npr. varnost) se ne bomo osredotočali. Na podlagi obstoječih rešitev primerov uporabe ter značilnosti poslovnih modelov določimo pomembnost posameznih kategorij, ki so prikazane na Sliki 3. Kot je predstavljeno v Poglavju 3.1 na primeru problematike trilema, so si določene konfliktne in je zato treba vedno iskati kompromise.

Najbolj dominantne kvantitativne nefunkcionalne zahteve opredelimo kot vhodne podatke priporočilnega sistema. Iz osnovnih predstavljenih kategorij bomo izbrali kvantitativne lastnosti, ki so pomembne za posamezni primer uporabe. Pri simulaciji bomo podali realistične ocene, da pridobimo rezultate kateri ustrezajo javnim EVM verigam blokom. Rezultati simulacije so predstavljeni v Tabeli 3 kjer se osredotočamo na nefunkcionalne kvalitativne zahteve značilne za uporabljene primere uporabe. Predvsem nas zanimajo stopnja distributivnosti (npr. število vozlišč, ki potrjujejo transakcije), hitrost transakcij v

korelaciji s hitrostjo generiranja blokov in cena transakcije. Slednja je zelo pomembna z vidika dolgoročne vzdržnosti, saj pri IoT primerih uporabe je število transakcij praviloma višje.

Algorithm 1 Algoritem večkriterijskega odločanja nefunkcionalnih zahtev

procedure VEČKRITERIJSKO ODLOČANJE

rezultati ← rezultat priporocilnega sistema nefunkcionalnih zahtev v obliki seznama EVM verig blokov
pogojniRezultati ← rezultat v obliki seznama EVM verig blokov kjer niso vse zahteve izpolnjene
neizpolnjeneZahteve ← seznam neizpolnjenih zahtev
zahteveUporabnika ← seznam vhodnih podatkov nefunkcionalnih zahtev v JSON obliki

while seznamzahteveUporabnikanipregledan **do** ▷ iteriramo skozi seznam nefunkcionalnih zahtev uporabnika

if kvantitativna zahteva **then**
▷ pregled ujemajočih se pogojev na podlagi kvantitativnih vrednosti iz obstoječih podatkov sistema in storitev umetne inteligence

else if kvalitativna zahteva **then**
▷ pregled ujemajočih se pogojev na podlagi kvalitativnih zahtev iz obstoječih podatkov sistema in storitev umetne inteligence

else if informativna zahteva **then**
▷ pregled pogojev informativne narave iz obstoječih podatkov in storitev umetne inteligence

end if
if neizpolnjena zahteva **then**

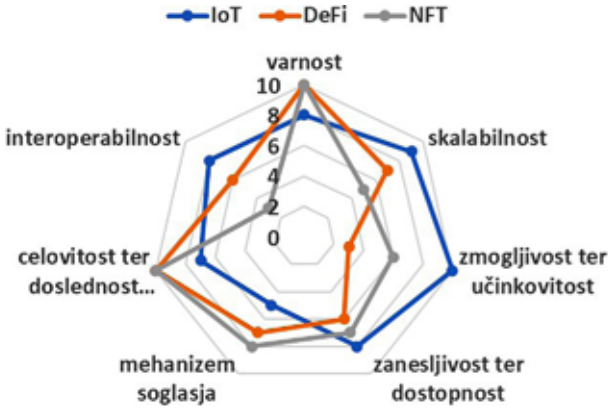
neizpolnjeneZahteve ← *trenutna zahteva*
pogojniRezultati ← *rezultati*
rezultati ← *null*
else

while seznamrezultatovnipregledanALIseznampogojniRezultatinipregledan **do**
▷ pregled ustreznosti zahteve z elementom rezultata (EVM veriga blokov) ▷ neustrezne elemente se odstrani iz seznama *rezultati* ALI *pogojniRezultati* ▷ v primeru, da je elementov v rezultati enako 0 se vse rezultate shrani v *pogojniRezultati* ter zahteve doda v *neizpolnjeneZahteve*
end while

end if
end while
end procedure

Tabela 1: **Osnovne plasti, ki opredeljujejo EVM verige blokov.**

Primer uporabe	Nefunkcionalne zahteve	Kvantitativne zahteve	Predlagane EVM verige blokov
IoT	Distributivnost, hitrost generiranja bloka, cena transakcije	nizka, minimalno 2 sekund, nizka	Optimism, Arbitrum, ShimmerEVM
DeFi	Distributivnost, hitrost generiranja bloka, cena transakcije, št. vozlišč	srednja, minimalno 17 sekund, srednja, visoko	Ethereum, Avalanche
NFT	Distributivnost, hitrost generiranja bloka, cena transakcije	visoka, minimalno 30 sekund, visoka	Binance Smart Chain, Polygon, Avalanche, Fantom



Slika 3: **Pomembnost nefunkcionalnih zahtev primerov uporabe.**

6 DISKUSIJA IN ZAKLJUČEK

Delo predstavlja analizo nefunkcionalnih zahtev, ki so smotrne za verige blokov s podporo EVM pametnih pogodb. Najprej smo predstavili razvoj nefunkcionalnih zahtev, ki je doseglo razpon s pojavom oblačnih arhitektur konec prvega desetletja tega tisočletja. Nato smo podrobneje pregledali razvoj plasti EVM verig blokov, ki so nastali kot odgovor na zahteve po podpori novih primerov uporabe. Pomemben aspekt našega dela predstavljajo javne storitve umetne inteligence, ki nam kot dodano vrednost nudijo informacije o EVM verigah blokov, saj obstoječi pristopi so težavni za dolgoročno vzdrževanje (npr. analiza preko API storitev, lastni spletni pajki idr., spletni viri z namenskimi informacijami idr.). Nefunkcionalne zahteve smo naslovili kot nadgradnjo obstoječega priporočilnega sistema, ki primarno kot rezultat vrača seznam predlog pametnih pogodb glede na podane funkcionalne zahteve sistema ter oblačne arhitekture.

V eksperimentalni študiji smo najprej izpostavili tri aktualne primere uporabe (IoT, DeFi in NFT). Najprej smo na podlagi orodij umetne inteligence ChatGPT in Microsoft Copilot identificirali najpomembnejše nefunkcionalne zahteve oziroma prisotne kot rezultat obeh orodij. Ocenili smo pomembnost nefunkcionalnih zahtev posameznih primerov uporabe ter ugotovili, da IoT primeru uporabe največjo težo dajejo hitrosti izvedbe pri čemer mehanizem soglasja igra najmanjšo vlogo. Pri primeru uporabe DeFi so najpomembnejša varnost, predvsem z vidika funkcionalnosti pametnih pogodb ter vsesplošna dovzetnost EVM verige blokov za napade (npr. premajhno število vozlišč verige blokov lahko vodi do t.i. 51% napada). Pri primeru uporabe NFT je varnost tudi dominantna lastnost, interoperabilnost je najmanj pomembna lastnost, saj praviloma se elementi transakcij ne prenašajo preko različnih verig in skalabilnost tudi ne igra pomembne vloge zaradi pretežno deterministčnega delovanja primera uporabe.

Glede na to, da je to prva različica sistema s podporo nefunkcionalnih zahtev, obstajajo precejšnje možnosti izboljšave. Kakovost informacij je mogoče izboljšati z bolj aktualnimi modeli za učenje orodij umetne inteligence, ki so prisotni v plačljivih različicah. Poleg tega bi bilo smiselno analizirati odgovore različnih storitev umetne inteligence z namenom izboljšave priporočilnega sistema. V prihodnje bi se lahko osredotočili na migracijo podatkov iz relacijske baze v obliko predstavitve podatkov ontologije, saj bi tako lahko vključili naprednejše pristope strojnega učenja ter pripravili naprednejša pravila sklepanja (angl. reasoning) na podlagi podatkov.

ZAHVALA

Raziskava je bila finančno podprta s sredstvi projekta Evropske unije Obzorje program za raziskave in inovacije na podlagi sporazumov št. 101093274 (TrustChain projekt: Fostering a Human-Centered, Trustworthy and Sustainable Internet) in št. 101092052 (BUILDCHAIN projekt: BUILDing knowledge book in the blockCHAIN distributed ledger. Trustworthy building life-cycle knowledge graph for sustainability and energy efficiency).

LITERATURA

[1] F. Liu, S. He, Z. Li, P. Xiang, J. Qi, and Z. Li, “An overview of blockchain efficient interaction technologies,”
[2] *Frontiers in Blockchain*, vol. 6, p. 996070, 2023.
[3] N. Szabo *et al.*, “Smart contracts: Building blocks for digital markets, extropy,” *The Journal of Transhu- manist Thought*, vol. 16, no. 18, pp. 2–20, 1996.
[3] M. Glinz, “On non-functional requirements,” in *15th IEEE international requirements engineering conference (RE 2007)*, pp. 21–26, IEEE, 2007.
[4] S. Nakamoto, “Re: Bitcoin p2p e-cash paper,” *The Cryptography Mailing List*, pp. 1–2, 2008.
[5] J. Yli-Huumo, D. Ko, S. Choi, S. Park, and K. Smolander, “Where is current research on blockchain technology?—a systematic review,” *PLOS ONE*, vol. 11, pp. 1–27, 10 2016.
[6] R. Nokhbeh Zaeem, K. C. Chang, T.-C. Huang, D. Liau, W. Song, A. Tyagi, M. Khalil, M. Lamison,
[7] S. Pandey, and K. S. Barber, “Blockchain-based self-sovereign identity: Survey, requirements, use- cases, and comparative study,” in *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, pp. 128–135, 2021.
[8] V. Buterin *et al.*, “Ethereum white paper,” *GitHub repository*, vol. 1, pp. 22–23, 2013.

[9] M. Staderini, E. Schiavone, and A. Bondavalli, "A requirements-driven methodology for the proper selection and configuration of blockchains," in *2018 IEEE 37th Symposium on Reliable Distributed Systems (SRDS)*, pp. 201–206, 2018.

[10] M. Kassab, "Exploring non-functional requirements for blockchain-oriented systems," in *2021 IEEE 29th International Requirements Engineering Conference Workshops (REW)*, pp. 216–219, 2021.
M. S. Farooq, M. Ahmed, and M. Emran, "A survey on blockchain acquainted software requirements engineering: Model, opportunities, challenges, and future directions," *IEEE Access*, vol. 10, pp. 48193– 48228, 2022.

[11] T. A. Tisha and M. M. A. Shibly, "Non-functional requirements for blockchain: Challenges and new directions," in *IOP Conference Series: Materials Science and Engineering*, vol. 1110, p. 012016, IOP Publishing, 2021.

[12] R. Jia and S. Yin, "To evm or not to evm: Blockchain compatibility and network effects," in *Proceedings of the 2022 ACM CCS Workshop on Decentralized Finance and Security*, DeFi'22, (New York, NY, USA), p. 23–29, Association for Computing Machinery, 2022.
S. Gec, D. Lavbič, V. Stankovski, and P. Kochovski, "Towards a smarter iot environment with ethereum virtual machine enabling ledgers," in *2022 International Conference on Computational Science and Computational Intelligence (CSCI)*, pp. 1229–1232, 2022.
G. Wood, "Polkadot: Vision for a heterogeneous multi-chain framework," *White Paper*, vol. 21, pp. 2327– 4662, 2016.

[13] S. K. Singh, S. Kumar, and P. S. Mehra, "Chat gpt google bard ai: A review," in *2023 International Conference on IoT, Communication and Automation Technology (ICICAT)*, pp. 1–6, 2023.

[14] Z. A. Mann, "Notions of architecture in fog computing," *Computing*, vol. 103, p. 51–73, jan 2021.

[15] S. Gec, V. Stankovski, D. Lavbič, and P. Kochovski, "A recommender system for robust smart contract template classification," *Sensors*, vol. 23, no. 2, 2023.

[15] S. Gec, P. Kochovski, V. Stankovski, and D. Lavbic, "Project oriented teaching approach of decentralised applications for undergraduate students," in *The 15th International Conference on Education Technology and Computers*, ICETC 2023, (New York, NY, USA), p. 207–215, Association for Computing Machinery, 2024.

▪

Sandi Gec je zaposlen kot asistent na Fakulteti za računalništvo in informatiko Univerze v Ljubljani, kjer pou- čuje predmete s področij informacijskih sistemov, spletnih tehnologij in decentraliziranih aplikacij. Dejaven je bil na številnih projektih, največ na mednarodnih v okviru Obzorje 2020 (npr. SWITCH, ENTICE, DECENTER, ONTOCHAIN idr.). Trenutno sodeluje na projektih TRUSTCHAIN in BUILDCHAIN v okviru Obzorje Evropa ter mednarodnem projektu ESSA, kjer se ukvarja z različnimi področji programskega inženirstva, predvsem z novimi pristopi tehnologije veriženja blokov s poudarkom na pametnih pogodbah ter interoperabilnost med različnimi verigami.

▪

Vlado Stankovski je redni profesor računalništva in informatike. Ima bogate izkušnje na področju program- skega inženirstva, računalništva v oblaku, na robu in v megli, porazdeljenih sistemov, semantike ter tehnologij umetne inteligence (strojno, globoko učenje). Sodeloval je pri načrtovanju, razvoju in integraciji tehnolo- gij vmesne programske opreme. Sodeloval je pri več nacionalnih in mednarodnih projektih, v konzorciju Superračunalniški center Slovenije, na projektu pametne specializacije IQ DOM ter v gruči za programsko inženirstvo projektov Obzorje 2020 kot predstavnik projektov ENTICE, SWITCH in DECENTER. Trenutno je v okviru Obzorje Evropa predstavnik projektov TRUSTCHAIN, BUILDCHAIN, ExtremeXP, EBSI-VECTOR in Swarmchestrate.