Pametni ugovori i nekretnine

Damir Hadžić
Softversko inženjerstvo i informacione tehnologije
Fakultet tehničkih nauka
Univerzitet u Novom Sadu
damirhadzic996@gmail.com

***Apstrakt*—Pametni ugovori zasnovani na *Blockchain*-u su izazvali veliki porast istraživačkog interesa zbog svog inovativnog načina funkcionisanja. Neke od karakteristika, kao što su: decentralizovano skladištenje transakcija, automatsko izvršavanje koda na decentralizovanoj mreži kao i pouzdana bezbednost čine ih pogodne za veliki broj primena pri automatizaciji procesa. Ovaj rad ima za cilj da uz pomoć *Blockchain* tehnologije, tačnije pametnih ugovora (engl. *Smart contract*), i *Ethereum* mreže demonstrira mogućnost kreiranja, skladištenja i izvršavanja međukoraka ugovora o kupoprodaji nepokretnosti. Kroz trivijalne funkcionalnosti programskog rešenja može se uvideti jednostavnost rada sa ugovorima iz perspektive korisnika. Kompleksnost rešenja se nalazi u komunikaciji sa decentralizovanom mrežom iz različitih nezavisnih slojeva sistema.**

Ključne reči— decentralizovano; ethereum; solidity; pametan ugovor; nekretnine;

I Uvod

U poslednjoj deceniji došlo je do velikog porasta interesovanja, kako za kriptovalute tako i za prvu među njima koja je značajno uticala na globalno ekonomiju, *Bitcoin*. Pored uloge koju je *Bitcoin* imao na ekonomiju, javlja se i druga sfera interesovanja. Tehnička pozadina iza kriptovaluta, sigurnosni protokoli kao i sve ostalo što donosi *Blockchain* tehnologija doveli su do nastanka novih ideja i širenja primene same tehnologije[1].

Bitcoin je kao prvi predstavnik proširio *Blockchain* tehnologiju i samim tim podstakao nastanak *Ethereum*-a kao sofisticiranije *Blockchain* platforme koja omogućuje komplikovane transakcije. Za razliku od svog prethodnika, koji je omogućavao transfer valute i *Bitcoin*-a, *Ethereum* nudi mogućnost da se koraci unutar transakcije, koja je već napravljena, izvršavaju u realnom vremenu bez zahtevanja posebne intervencije ili posrednika[2].

Kako su same transakcije znatno kompleksnije na *Ethereum* platformi, razvijeno je posebno okruženje za izvršavanje ovakvih transakcija, kao i jezik pomoću kojeg je omogućeno pisanje kompleksnih međukoraka transakcije, tačnije *Solidity*[3]. Prvi put se uvodi termin pametnog ugovora, kao inovativne ideje *Ethereum* platforme. Ovakvi ugovori predstavljaju jedinice koje se skladište na distribuiranoj mreži, čuvaju podatke vezane za svoju instancu (engl. *instance*) i omogućuju izvršavanje prethodno napisanih funkcionalnosti koje menjaju stanje ugovora ili dobavljaju podatke iz samog ugovora. Funkcionalnosti koje predstavljaju sam deo pametnog ugovora je moguće napisati po zahtevima same svrhe koju je potrebno postići, a fleksibilnost u pisanju pametnog ugovora pruža *Solidity* programski jezik.

Kako su počele da se istražuju nove primene *Ethereum* mreže van dometa kriptovaluta, uočena je mogućnost primene pametnih ugovora u pravu, tačnije baš kao zamena za dosadašnje pravne ugovore. Pametan ugovor iz ugla pravnog ugovora bi predstavljao samoizvršavajući ugovor sa dogovornim tačkama između stranaka koje predstavljaju ugovorene strane ugovora. Konkretnije, za ugovore o kupoprodaji nekretnine, kupac i prodavac ne zahtevaju postojanje dodatnog lica kako bi izvršili transfer nekretnine sa jednog lica na drugo. Pametan ugovor sadrži unapred isprogramirana pravila, tako da kada podaci koje ugovor sadrži podlegnu promenama, pokreću se izmene nad ugovorom ukoliko su zadovoljeni kriterijumi za njihovo izvršavanje[4].

II Pregled tehnologija

U poglavljima ove sekcije biće dat osvrt na tehnologije koje stoje iza implementacije softverskog rešenja ovog rada. Za početak će biti opisan sam *Ethereum Blockchain*, nakon toga nešto detaljnije šta je tačno pametan ugovor, a potom i način komunikacije sa distribuiranom mrežom.

II.A Ethereum Blockchain

Ethereum Blockchain je vrste tehnologije distribuiranih knjiga (engl. *distributed ledger technology* (*DLT*))[5], termin koji se generalno koristi za baze podataka koje čuvaju informacije u decentralizovanim mrežama nezavisnih čvorova.

Predstavlja platformu zasnovanu na *Blockchain* tehnologiji, otvorenog koda (engl. *open-source*) koja otklanja potrebu za postojanjem posrednika. Po izjavama Dr. Gavin Wood-a, suosnivača *Ethereum* organizacije, *Ethereum* kao celina može da se posmatra kao velik uređaj stanja (engl. *state machine*) zasnovan na transakcijama[6]. Ovakva distribuirana mreža omogućuje svima da postave svoja pravila za izvršavanje transakcija kao i uslove za prelazak u naredna stanja.

Ono što je *Ethereum* unapredio u odnosu na osnovnu *Bitcoin* verziju *Blockchain*-a, jeste mogućnost pisanja dodatne logike koja se izvršava prilikom dovođenja jedne jedinice izvršnog bloka u određeno stanje. Svaki od ovih međukoraka koji se izvršavaju predstavljaju transakciju unutar distribuirane mreže. Naravno, kako bi sistem bio održiv, svaka od ovih

transakcija mora da ima takozvani trošak (engl. *fee*) koji je potrebno platiti da bi se ista izvršila.

II.B Pametan ugovori

Kao jedna od funkcionalnosti *Blockchain*-a koja najviše obećava, u poslednjih par godina je privukla pažnju različitim sektorima, kao što je u primeru ovog projekta pravo. Pametni ugovori su u osnovi mali programi koji se skladište na *Blockchain*-u. Ovo ih čini neizmenjivim (engl. *immutable*) dok god integritet *Blockchain*-a nije kompromitovan. S tim na umu, krajnji korisnik može biti siguran da program nije menjan niti manipulisan. Ovakvo ponašanje gradi u ljudima poverenje u distribuirane sisteme, što bi značilo da ne postoji potreba da se veruje nekom trećem licu odnosno posredniku. Ovakve vrste programa nalaze primenu u velikom broju procesa koji zahtevaju postojanje posrednika, odnosno ovlašćenog trećeg lica.

Pošto je svaka isporuka novog ugovora proces koji podatke smešta na distribuiranu mrežu, dovodi se u pitanje kako se ponovo pristupa podacima ugovora kao i samom ugovoru. Prilikom isporučivanja ugovora na udaljenu mrežu, kao odgovor od mreže se dobija status same isporuke. Ukoliko je isporuka uspešno izvršena, u odgovoru se nalazi adresa tog ugovora na mreži, koje je predstavljena kao jedinstvena adresa sačinjena od 42 karaktera u heksadecimalnom zapisu, pri čemu prva dva simbola predstavljaju oznaku da se radi o heksadecimalnom zapisu. Pomoću ove adrese je u svakom trenutno omogućeno pristupa podacima ugovora na *Ethereum* mreži kao i poziv funkcionalnosti nad samim ugovorom.

Ethereum omogućuje programerima to napišu svoje pametne ugovore. Svaki od tih ugovora ima mogućnost da:

- Funkcioniše kao više-potpisni nalog (engl. *mutli-signature account*), odnosno da trošenje sredstava unutar ugovora zahteva odobrenje od većeg broja ljudi
- Upravlja uspostavljenim dogovorima između korisnika, npr. prepisivanje nepokretnosti sa jednog lica na drugo prilikom plaćanja nepokretnosti
- Pruža podršku drugim ugovorima, na način kao što aplikacije prilikom izgradnje koriste programske biblioteke
- Skladišti informacije o aplikaciji i korisnicima

II.C Komunikacija sa mrežom

U ovom delu teksta biće objašnjeno na koji način se vrši uspostavljanje konekcije ka distribuiranoj *Ethereum* mreži, pristup za testiranje pametnih ugovora u virtualnom okruženju koje podržava *Solidity* programski jezik, kao i različite ideje koje su testirane tokom implementiranja softverskog rešenja ovog rada.

Prvi korak u fazi implementacije bio je pronalaženje okruženja, tačnije distribuirane mreže koja omogućuje isporučivanje ugovora i izvršavanje transakcija. Ideja je bila da se na *TestNET Ethereum* mreži isporučuju pametni ugovori. Ovaj tip mreže je napravljen kao simulacija stvarne mreže, gde su sva sredstva bez prave vrednosti, iako su potrebna za plaćanje troškova transakcija. Na takvoj mreži je bilo potrebno napraviti elektronski novčanik kako bi se mogao testirati ugovor tokom razvoja, kao i da se koristi takav novčanik prilikom stvarnog rada aplikacije. Problem je predstavljalo pronalaženje sredstava, tačnije novčanika koji skladišti dovoljno sredstava kako bi se mogli isporučivati ugovori i izvršavati njegove transakcije.

S obzirom na komplikacije prilikom pronalaženja takvog novčanika, pronađen je alat koji omogućuje pokretanje lokalne *Ethereum* mreže, gde je moguće alocirati potreban broj naloga, novčanika kao i sredstava na svakom novčaniku. Naziv ovog alata je *Ganache*, i omogućio je pokretanje servera jednostavnom instalacijom i pokretanjem *Node* paketa *ganache-cli*.

Nakon što je problem servera koji izvršava transakcije pametnih ugovora rešen, potrebno je bilo pronaći *provider*-a koji će obezbediti konekciju i komunikaciju sa serverom gde ste to izvršava. *Web3* predstavlja *provider* koji nudi jednostavnu implementaciju za rad sa udaljenom mrežom pomoću *Node* paketa *web3js*. Sa druge strane, testiranje i razvoj *Solidity* koda nije jednostavan, i pored ovog *provider*-a bilo je potrebno pronaći način za ubrzan razvoj i testiranje pametnih ugovora o kupoprodaji nekretnina. *Remix IDE* predstavlja sofisticirano rešenje koje olakšava razvijanje pametnih ugovora.

Veb aplikacija *Remix IDE* nudi integrisan *Web3* provider koji je moguće povezati na bilo koju mrežu ukoliko se proslede potrebni poverljivi podaci. U slučaju ovog rada, povezivanje je vršeno na lokalnu *Ethereum* mrežu, i nije bilo potrebno pružanje kredencijala za povezivanje. Kroz svoj dobro dizajniran interfejs, nudi znatno olakšan rad sa isporukom pametnih ugovora kao i testiranjem njegovih metoda.

III Struktura ugovora

Za konstrukciju ugovora koji je čitljiv čoveku i pravno validan, potreban je određen skup podatak koji je promenjiv u zavisnosti od stranaka koje vrše sklapanje ugovora. S tim u vidu, kako bi se jasnije objasnile funkcionalnosti ugovora, potrebno je prvo osvrnuti se na podatke koje je potrebno uneti i skladištiti unutar pametnog ugovora. Obavezni podaci pametnog ugovora predstavljaju:

- Podaci o kupcu (ime i prezime, adresu stanovanja, grad stanovanja, jedinstveni matični broj građana, MUP registrovane lične karte)
- Podaci o prodavcu (ime i prezime, adresu stanovanja, grad stanovanja, jedinstveni matični broj građana, MUP registrovane lične karte)
- Podaci o nepokretnosti (identifikacioni broj iz lista nepokretnosti, adresa nepokretnosti, površina, osnovna cena, osnovna cena slovima)
- Podaci o samom ugovoru (datum sklapanja ugovora, adresa sklapanja ugovora, nadležni sud, osoba zadužena za plaćanje troškova ugovora)

Pored gorenavedenih podataka, moguće je skladištenje podataka koje krajnji korisnik ima mogućnost da doda po zahtevu. Ovi podaci se odnose na opcione članove ugovora, i u njih spadaju:

- Podaci o posredniku (ime i prezime, jedinstveni matični)
- Podaci o depozitu (iznos depozita, iznos depozita slovima)
- Podaci o plaćanju u delovima (broj delova)
- Podaci o datumu napuštanja nepokretnosti (datum napuštanja)
- Podaci o plaćanju režija

Za pristup svakom od ovih podataka definisane su metode unutar pametnog ugovora. Metoda za čitanje (engl. *getter*) kao i metoda za pisanje (engl. *setter*) svakog podatka predstavljaju osnovu za rad sa ugovorom. Pored ovih metoda, postoji i specifična metoda, koja služi za proveru da li je ugovor došao u finalno stanje, tačnije da li su svi potrebni troškovi realizovani. Ovaj metod se menja u zavisnosti od opcionih polja, koji dodatno proširuju logiku metode.

IV Funkcionalnosti

Kako bi se uvidela primena kreiranja ugovora o kupoprodaji nepokretnosti koristeći pametne ugovore, implementiran je skup funkcionalnosti sa podelom na dve grupe. Prvi deo tih funkcionalnosti bavi se radom sa udaljenom *Ethereum* mrežom, dok preostali skup funkcionalnosti podržava zahteve iz realnog skupa zahteva u radu sa ugovorima o kupoprodaji nepokretnosti.

Funkcionalnosti za rad sa realnim skupom zahteva:

- Popunjavanje forme podacima koji su potrebni za ugovor o kupoprodaji nepokretnosti
- Dodavanje opcionih delova ugovora po potrebnim
- Prikaz izvornog koda pametnog ugovora u *Solidity* programskom jeziku (reflektujući promene u zavisnosti od modifikacije)
- Prikaz ugovora čitljivog čoveku (reflektujući promene u zavisnosti od modifikacije, kao i unete podatke)
- Kreiranje ugovora
- Prikaz već isporučenih ugovora, tačnije čitanje podataka tih govora sa distribuirane mreže

Funkcionalnosti za rad sa decentralizovanom mrežom:

- Modifikaciju ugovora pisanog u *Solidity* programskom jeziku
- Kompajliranje (engl. *compiling*) modifikovanog ugovora pomoću *Solidity* biblioteke
- Isporuka ugovora sa odgovarajućim podacima na decentralizovanu mrežu
- Poziv metoda ugovora (postavljanje vrednosti, dobavljanje vrednosti, provera da li je ugovor u izvršenom stanju)

IV.A Popunjavanje forme

Svaki ugovor o kupoprodaji nepokretnosti zahteva obavezne podatke kao što su: osnovni podaci o kupcu, osnovni podaci o prodavcu, broj nepokretnosti iz liste nepokretnosti, površinu nepokretnosti, adresa nepokretnosti, cena kupoprodaje. Pored ovih podataka, forma zahteva i podatke specifične za sam ugovor: adresa mesta zaključivanja ugovora, datum zaključivanja ugovora, sud kao pravni predstavnik ugovora, nosioca troškova potpisivanja ugovora.

Opciono, korisniku se nude dodatni članovi kojima je moguće proširiti i dalje specificirati obavezujuće stavke ugovora. Pod dodatne stavke ugovora spadaju: vršenje kupoprodaje preko posrednika, plaćanje depozita od strane kupca, plaćanje nepokretnosti u delovima, krajnji datum napuštanja nepokretnosti kao i plaćanje računa od strane prodavca pre napuštanja predmeta kupoprodaje.

IV.B Prikaz izvornog koda

Provera ispravnosti pametnog ugovora od strane tehničkog lica može se vršiti pregledom izvornog koda. Nakon unosa mandatornih kao i opcionih polja, vrši se konstruisanje izvornog koda pametnog ugovora pisanog u *Solidity* programskom jeziku. Obavezna polja ne menjaju sam pametni ugovor u smislu opsega funkcionalnosti koje pruža. Međutim, uključivanje opcionih polja u ugovor proširuje funkcionalnosti pametnog ugovora, i samim tim dodaje linije koda u izvorni kod. Svako uključeno opciono polje reflektovano je dodavanjem međusobno nezavisnih delova koda te je time omogućena uključivanje stavki ugovora u bilo kojoj kombinaciji.

IV.C Prikaz ugovora u formatu čitljivom čoveku

Svi učesnici ugovora pomoću ove funkcionalnosti imaju na uvid ugovor u tradicionalnom formatu čitljivom čoveku. Identično kao u prethodno opisanoj funkcionalnosti, uneta mandatorna polja ne menjaju izgled ugovora već su neophodna za generisanje istog. Svako od opcionih polja dodaje novi član, gde se nalazi propratni tekst u zavisnosti od samog tipa opcionog člana.

IV.D Kompajliranje pametnog ugovora

Kako bi se pametan ugovor isporučio na *Ethereum* mrežu, potrebno je da prvobitno bude napisan na *Solidity* jeziku, u skladu sa zahtevima zadatim virtualnim okruženjem u kojem se izvršava. Svaka od mogućih kombinacija odabira opcionih polja kao proizvod daje različit finalni programski kod ugovora. Faza koja prethodi isporučivanju ugovora na distribuiranu mrežu je kompajliranje koda radi kreiranja šeme ugovora (stručni termin ABI) kao i provere validnosti napisanog koda ugovora. Za ovaj korak se koristi *Node* paket pod nazivom *solc*, koji sadrži Solidity kompajler (engl. *compiler*) sa svega par funkcionalnosti koje omogućuju kompajliranje koda, prijavljivanje greške u kodu kao i generisanje šeme distribuiranog ugovora.

IV.E Isporuka ugovora

Nakon uspešno popunjene forme regulisane obaveznim poljima i validatorima, korisniku se omogućuje isporuka ugovora na veoma trivijalan način. Pošto se u korake isporuke ubraja i korak kompajliranja kao i komunikacije sa *Ethereum* mrežom, sam zahtev iziskuje mali odziv kako bi se izvršio. Ukoliko uspostavljanje konekcije sa udaljenom mrežom, kompajliranje kao i sam proces isporuke prođe bez problema,

korisniku se na vrhu ekrana prikazuje poruka sa adresom tog ugovora na *Ethereum* mreži. Takođe, korisnik je preusmeren na početnu stranu aplikacije gde ima uvid u sve prethodno isporučene pametne ugovore, među kojima se sada nalazi i upravo isporučen ugovor, na dnu liste.

U slučaju greške na mreži ili greške unutar koda softverskog rešenja, korisniku se prikazuje poruka da je ugovor nije uspešno isporučen, i ne dolazi do preusmerenja.

IV.F Poziv metoda ugovora

Rad sa ugovorom kao i izmene stanja ugovoru omogućene su definisanjem metoda ugovora, odnosno pisanjem funkcionalnosti ugovora u *Solidity* programskom jeziku. Svaki od poziva ka metodi ugovora koji se nalazi na distribuiranoj mreži predstavlja jednu transakciju unutar mreže. Shodno tome, u zavisnosti od kompleksnosti definisane metode povećava se i trošak koji je potrebno platiti za izvršavanje te metode.

Kako je prilikom isporuke moguće postavljanje samo određenog skupa podataka (ograničen broj parametara koji se prosleđuju prilikom kreiranja instance ugovora), metode nad pametnim ugovorom o kupoprodaji nepokretnosti se mogu podeliti na dve grupe:

- metode pregleda (engl. *view functions*)
- metode izmene (engl. *alter state functions*)

Kako bi se izvršila inicijalizacija opcionih podataka prilikom isporuke ugovora, vrši se poziv metoda za pisanje podataka za koje je to potrebno.

Prilikom pregleda podataka ugovora, potrebno je izvršiti metode za pregled podataka. Ovaj korak je trivijalan za mandatorna polja, jer ona uvek postoje. Prilikom poziva metode za pregled podataka u slučaju opcionih polja, na *Ethereum* mreži se za izvršavanje ovakvih transakcija dešava greška. Greška unutar ovih metoda predstavlja nepostojanje opcionih podataka tog tipa, i samim tim predstavlja indikator da taj tip opcionih podataka na postoji nad određenim ugovorom. Ovim mehanizmom je obezbeđeno dobavljanje svih podataka koji postoje unutar ugovora.

Pored ova dva tipa metoda, postoji i metod koji proverava da li se pametan ugovor nalazi u finalnom stanju, tačnije da su ispunjeni svi uslovi ugovora kako bi se izvršio transfer nepokretnosti. Metod je napisan kako bi se ostavio prostor za proširivanje softverskog rešenja. Iako je testiran u development okruženju, u ovoj aplikaciji ne postoji poziv ka toj metodi.

V Slojevi arhitekture

Kako bi se bolje razumela tehnička strana sistema koji radi sa pametnim ugovorima zasnovanim na ugovorima o kupoprodaji nepokretnosti, u ovom poglavlju su opisani slojevi sistema, kao i njihove uloge i funkcionalnosti koji nude.

Opis će teći od najnižih slojeva, tačnije od same *Ethereum* distribuirane mreže, ka sloju poslovne logike pa sve do aplikativnog sloja tačnije sloja koji vidi krajnji korisnik.

V.A Ethereum mreža

Kako bi se omogućilo pokretanje sloja poslovne logike, prvo je potrebno pokrenuti *Ethereum* mrežu, odnosno u slučaju ovog rešenja lokalni *GanacheCLI* server. Server se pokreće sa jednim nalogom za koji je dodeljeno 100 *Ethereum* kriptovaluta, što predstavlja sredstva za vršenje transakcija. Pored toga, postavlja se i poseban uslov (engl. *flag*) kako bi se omogućilo pravljenje trenutno napisanog ugovora u *Solidity* programskom jeziku. U slučaju rada na nekoj od udaljenih distribuiranih mreža, potrebne bi bile manje izmene nad ugovorom kako bi se smanjili troškovi kreiranje ugovora. Kako fokus ovog rada nije bila optimizacija za udaljene mreže, ovaj korak je preskočen i omogućeno je neoptimizovano pravljenje pametnih ugovora (neoptimizovano u smislu trošenja sredstava, što u lokalnoj mreži ne predstavlja problem).

V.B Sloj poslovne logike

U sloju poslovne logike vrši se uspostavljanje konekcije ka lokalnoj *Ethereum* mreži, kompajliranje ugovora, vraćanje izvornog koda ugovora, isporučivanje ugovora na distribuiranu mrežu kao i skladištenje i dobavljanje prethodno isporučenih ugovora. Za pokretanje sloja poslovne logike potrebno je imate prethodno aktivan *GanacheCLI* server, koji zauzima port 8545, što je ujedno i standardni port koji taj server zauzima.

Korisniku aplikativnog sloja se na korišćenje nude tri funkcionalnosti nad kojima je moguće vršenje poziva, a to su: isporučivanje ugovora, dobavljanje izvornog koda ugovora i dobavljanje adresa već isporučenih ugovora.

V.C Aplikativni sloj

Korisnički interfejs predstavljaju dve stranice, od kojih jedna služi za pregled specifičnog ugovora (pri čemu se na istoj stranici sa strane nalazi lista svih prethodno isporučenih ugovora) dok druga stranica sadrži formu za unos podataka potrebnih za kreiranje ugovora. Nakon uspešno unetih mandatornih polja, kao i dodatih opcionih delova ugovora, korisnik pored funkcionalnosti za isporuku ugovora ima opciju i pregleda ugovora. Prilikom poziva ove funkcionalnosti, korisnik na uvid dobija ugovor napisan u *Solidity* program jeziku sa jedne strane prozora, i ugovor napisan u ljudski čitljivom obliku sa druge strane prozora.

Nakon uspešnog kreiranja ugovora, korisniku se prikazuje poruka sa adresom tog ugovora i vrši se usmerenje na stranicu na pregled ugovora. Prilikom neuspešne komunikacije sa aplikativnim slojem, korisnik dobija poruku o statusu greške.

VI Zaključak

Brže izvršavanje, smanjen rizik i transakcije visoke bezbednosti obezbeđene hešovanjem (engl. *hashing*) učinile su *Blockchain* veoma jakom tehnologijom. Nemogućnosti izmene (engl. *immutability*) blokova kao i par principa dokaz o radu (engl. *proof of work*) i dokaz o autoritetu (engl. *proof of authority*) čine ovu tehnologijom otpornom na prevare[7]. Po statistikama iz ekonomije, u eri digitalne prodaje nekretnina dešava se velik broj prevara.

Sa nailaskom novih tehnologija kao što su kriptovalute i *Blockchain*, javlja se i šansa da blagovremeno ugovori, kao i papirni novac, bude smenjen kriptovalutama i pametnim

ugovorima koji pružaju bezbedno i decentralizovano okruženje za skladištenje valuta kao i izvršavanje transakcija. Ovaj rad je pružio širu sliku o mogućnostima pametnih ugovora za rešavanja procesa u pravnim sistemima. Pored osnovne ideje koja je demonstrirana, ostavljen je prostor za dodatna proširenja softverskog rešenja.

Kako se rad sa ugovorom vrši na lokalno podignutoj mreži sa proizvoljno alociranim resursima za plaćanje, prvi korak za unapređenje bi bio rad na *TestNET Ethereum* okruženju. Prelazak na takvo rešenje ne zahteva mnogo vremena, potrebno je alocirati novčanik sa dovoljno sredstava kao i nalog na *TestNET* mreži, dok celokupna implementacija, osim uspostavljanja konekcije ka mreži, ostaje netaknuta. Nakon prelaska na distribuiranu mrežu, moguća su proširenja u vidu funkcionalnosti. Neke od mogućih funkcionalnosti su:

- Implementacija interfejsa za poziv metoda koje menjaju stanje ugovora kao i implementacija samog poziva tih metoda
- Implementacija interfejsa za proveru stanja ugovora
- Registracija korisnika i automatsko popunjavanje ugovora pomoću podataka korisnika
- Plaćanje nepokretnosti unutar same *Ethereum* mreže, tačnije u *Ethereum* valuti
- Razdvajanje novčanika svakog od korisnika od novčanika koji se koristi za kreiranje pametnih ugovora. U sklopu ovog koraka, bilo bi potrebno vršiti transfer sredstava sa jednog novčanika na drugi u zavisnosti od realnog zahteva pravnog ugovora

Poslednji korak nakon celokupne validacije sistema predstavljao bi prelazak na *MainNET Ethereum* mrežu, odnosno realan sistem na koje se trenutno nalaze sredstva ove distribuiranje mreže.

Reference

1 Pushpit Bhardwaj, Yuvraj Chandra, Deepesh Sagar, „Ethereum Data Analytics: Exploring the Ethereum Blockchain“, Department of Computer Science, Shaheed Sukhdev College of Business Studies, University of Delhi 18 September 2021

2 Andrea Janssen, „The Formation of Smart Contracts and Beyond: Shaking the Fundamentals of Contract Law?“, Radbound University 2018

3 Carlos, M.E., I. Solaima, I. Sfyrakis, J. Ng-Crowcroft. “Implementation of Smart Contracts Using Hybrid Architectures with On- and Off-Blockchain”, 2018

4 Ph.D. Researcher, Mohamed Saleh Darwish, „Applicability of Smart Contracts for Real Estate: A literature review“, Article in International Journal of Scientific and Engineering Research · November 2021.

5 Dr. Gavin Wood, „Ethereum: A secure decentralised generalised transaction ledger“, 07.01.2022.

6 Pushpit Bhardwaj, Yuvraj Chandra, Deepesh Sagar, „Ethereum Data Analytics: Exploring the Ethereum Blockchain“, College of Business Studies, University of Delhi 18 September 2021

7 Bharambe Asha, Motwani Ruchika, Rathi Abhishek, Kothari Parth, „Smart Contract for Real Estate Using Blockchain“, Dept. of Information Technology April 2020.