Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.41
DOI: https://doi.org/10.24867/16BE09Kovacevic

ПРИМЕНА *HYPERLEDGER FABRIC BLOCKCHAIN*-А У ПОСЛОВНИМ ПРОЦЕСИМА

APPLICATION OF THE *HYPERLEDGER* FABRIC BLOCKCHAIN IN BUSSINESS PROCESSES

Ивана Ковачевић, *Факултет техничких наука, Нови Сад*

Област – ЕЛЕКТРОТЕХНИКА И РАЧУНАРСТВО

Кратак садржај – *Овај рад наводи теоријске основе на којима је заснована blockchain технологија. Објашњава концепте Hyperledger Fabric blockchain технологије. У раду је описан модел имплементираног решења и представљене су основне функционалности система које се ослањају на blockchain мрежу.*

Кључне речи: *blockchain, Hyperledger, паметни уговор, концезус алгоритам*

Abstract – *This paper presents theoretical basis of blockchain technology. The study includes concepts of Hyperledger Fabric blockchain and explains major considerations for implementation of system for applying for master thesis.*

Keywords: *blockchain, Hyperledger, smart contract, consensus algorithm*

1. УВОД

Пословни процеси имају за циљ оптимизацију послова како би се обезбедиле боље перформансе пословних система, што укључује повећање профита из угла компанија, и веће задовољство пруженим услугама, уз угла потрошача. Критичну тачку у пословним процесима представља верификација и трансформација дигиталних средстава који учествују у комуникацији између компанија без оствареног међусобног поверења.

Како би се постојећи проблем поверења превазишао, пословни процеси могу да се реализују ослањајући се на *blockchain* технологију и њене особине децентрализације и непроменљивости података. Овај рад се базира на примени *blockchain* технологије на пословним процесу у области високог образовања. Ова технологија у себи садржи криптографске примитиве које би, уколико се имплементирају на адекватан начин, могле значајно да допринесу смањењу плагираних или фалсификованих радова и диплома, што представља један од главних проблема са којима се сусреће академска заједница. Особина дистрибуираности *blockchain* мреже олакшава и убзава поступак провере исправности докумената, јер не захтева од заинтересованих страна да се директно обрате централизованом телу.

НАПОМЕНА:
Овај рад је проистекао из мастер рада чији ментор је био др Горан Сладић, ред. проф.

2. *HYPERLEDGER FABRIC*

Hyperledger Fabric [1] је радни оквир за развој дистрибуираних приватних *ledger*-а. Представња основу за развој апликација и решења са модуларном архитектуром. *Hyperledger Fabric* омогућава да компоненте као што су паметни уговори, концензуси или сервис за чланство (енгл. *membership service*) буду плагабилни, односно да их је могуће заменити или додати у систем по потреби.

2.1. *Hyperledger Fabric* модел

Hyperledger Fabric се састоји од асета, концензуса, дељеног *ledger*-а, паметног уговора и сервиса за чланство. *Ledger* чува тренутно стање пословних система у форми дневника трансакција. Физичке компоненте *Hyperledger Fabric-a* су чворови и с обзиром да је мрежа приватна, они могу међусобно да формирају канале за комуникацију скривену од остатка мреже. Приступањем каналу, компоненте прихватају међусобну сарадњу и деле идентичне копије *ledger-a* повезане са тим каналом. Канали су структуре на логичком нивоу, док чворови представљају контролне тачке за приступ и управљање каналима. Унутар *Hyperledger Fabric*-а могуће је приступати средствима (енгл. *asset*) или их мењати користећи трансакције паметних уговора. Средства на мрежи се представљају као колекција кључ-вредност парова, и могу да се чувају у бинарном или JSON формату.

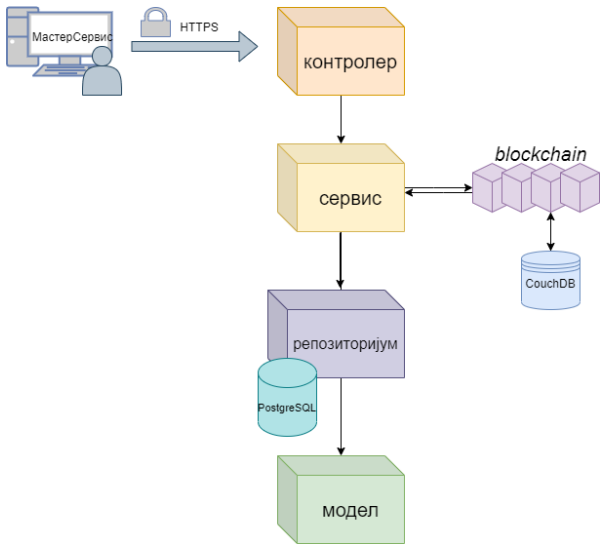
2.2. Паметни уговори и постизање концензуса

Паметни уговори представљају скуп договорених правила која чине пословни модел интеракције између организација [2]. У *Hyperledger Fabric* мрежи, термин који се користи за паметне уговоре је *chaincode*. *Chaincode* представља трансакциону логику која контролише животни век објеката на стању света. Са сваким паметним уговором је повезана и политика одобравања (енгл. e*ndorsement policy*). Политика одобравања дефинише које организације у *blockchain* мрежи морају да потпишу трансакцију како би се трансакција сматрала валидном [3]. *Chaincode* се пише у једном од тренутно подржаних програмских језика, који имплементирају прописани интерфејс. *Chaincode* се извршава у заштићеном *Docker* [4] контејнеру који је изолован од *endorsing* чворова. Стање на *ledger*-у које је креирано и ажурирано од стране једног *chaincode*-а је доступно само том *chaincode-y*. Концензус представља једну или скуп функција које примарно служе да би се постигао договор око редоследа трансакција. У *Hyperledger Fabric* технологији, концензус се постиже

када *orderer* и резултати трансакција једног блока испуњавају све критеријуме провере. Ове провере се одвијају током животног века трансакције, и укључују политику одобрења како би се тачно одредило који чланови мреже одобравају одређени тип трансакције.

3. МОДЕЛ СИСТЕМА

Намена система јесте да обезбеди информациону инфраструктуру за поступак пријаве и одбране завршних мастер радова на факултету. Систем је развијен као *web* апликација, са одвојеним клијентским и серверским делом, који остварују међусобну комуникацију употребом HTTPS протокола. Клијентска апликација се извршава у прегледачу (енгл. *browser*), док је серверски део монолитна апликација која се састоји од пакета, тако да подржава вишеслојну архитектуру. Серверски део комуницира са *Hyperledger Fabric SDK*-ом како би сместио и преузео потребне податке на *blockchain* мрежу. Архитектура система је приказана на слици 1.

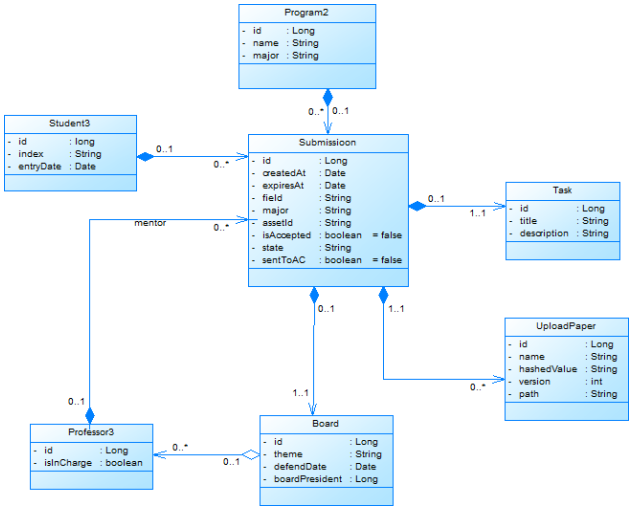


Слика 1. *Архитектура система*

Сви корисници система су део исте факултетске мреже, а систем сам по себи не пружа функционалности које би требало да буду видљиве неаутентификованим корисницима.

Примарни корисници система за предају мастер радова су студенти. Поред студената, апликацију користе још и референти студентске службе, чланови наставно научног већа, руководилац студијског програма и професори. Централно место система представља поступак подношења захтева за израду мастер рада и класе којеу учествују у овом процесу су дате на дијаграму на слици 2. Класа *Submission* моделује захтев који студент попуњава приликом пријаве за израду мастер рада, и поред основних поља за пријаву, садржи и додатна поља која прате статус у ком се пријава тренутно налази.

Пошто се пријава складишти и на *blockchain*-у и у бази података, постоје два јединствена идентификатора у класи, *id* који је идентификатор за базу података, и *assetId*, идентификатор који се користи приликом приступа *blockchain*-у. Идентификатор пријаве на *blockchain*-у ће доћи на систем као одговор приликом успешног додавања нове пријаве. Поред тога, за сваку пријаву се бележи датум када је настала и до кад важи, затим област из које студент жели да пише мастер рад, студијска група/подгрупа којој припада студијски програм на који је студент уписан као и додатни подаци о студенту и професору.



Слика 2. *Класни дијаграм за пријаву мастер рада*

4. ИМПЛЕМЕНТАЦИЈА

Blockchain мрежа је конфигурисана ослањајући се на *test-network* пример који је преузет са званичног git налога *Hyperledger Fabric*-а. Након што се пример преузме, у креираном фолдеру се налазе сви потребни CLI бинарни фајлови и докер слике потребне за конфигурацију и покретање мреже. За потребе система пријаве мастер рада, формиране су три организације, свака са по два чвора и један *ordering* сервис.

У овом решењу организације су подељене тако да прву организацију чине студенти, другу организацију представља студентска служба а трећу организацију чине професори. У решењу постоји један *ordering* сервис који управља трансакцијама, што се може оправдати чињеницом да поступак пријаве мастер рада није високо фреквентна операција, те *ordering* сервис неће бити оптерећен бројем пристиглих трансакција. Мрежа садржи два канала, ради очувања безбедности података који се налазе у трансакцијама. Један канал се користи за комуникацију између студената и студентске службе, а други канал преноси трансакције између студентске службе и професора. За генерисање сертификата се користио *Fabric CA*.

4.1 Регистрација корисника и генерисање сертификата

Прва компонента која се поставља на мрежу је сертификационо тело (CA). Ово тело служи за генерисање и потписивање осталих сертификата у мрежи, као што су сертификати везани за чворове и сертификати који идентификују администраторе и кориснике чворова. За регистрацију и упис администратора и корисника чворова, користи се *Fabric CA* клијент.

Клијент ће прво регистровати чворове, потом кориснике и на крају администратора организације.

Приликом регистровања, дефинише се којој организацији ти учесници припадају. Затим, сваки регистровани учесник има свој идентификатор у оквиру организације и тајну или лозинку. Такође, за сваког корисника се дефинише тип. Дефинисани типови су чвор, клијент и администратор. За TLS комуникацију између организација, наводи се путања до TLS сертификата.

Након што се корисници региструју, потребно је генерисати MSP фолдере за сваки чвор. За генерисање MSP фолдера се користи *enroll* команда при чему је потребно навести на ком порту је организација којој чвор припада подигнута.

4.2 Креирање чворова и канала

Параметри везани за конфигурацију чвора се налазе у `core.yaml` фајлу. Битни параметри који се конфигуришу у овом фајлу су идентификатор и назив чвора, мрежа којој чвор припада, адреса на којој ће чвор да "ослушкује" захтеве, путања до MSP фолдера као и подаци везани за *chaincode*. Сваки чвор комуницира са остатком мреже помоћу *gossip* протокола. Како би се смањио проток саобраћаја кроз канал, *Hyperledger Fabric 2.0* је конфигурисан тако да чворови преузимају нове блокове директно од *ordering* сервиса, уместо од других чворова [5].

У систему који је имплементиран, постоји само један *orderer* при чему се он ставља на располагање организацијама приликом креирања канала. За креирање *orderer*-а важе исти предуслови као и за креирање обичног чвора. Сваки *orderer* треба да има свој сертификат и приватни кључ који ће користити за потписивање трансакција, затим да има TLS сертификате као и MSP фолдер организације.

Канали се креирају тако што се прво креира трансакција за креирање канала у *genesis* блоку, а потом се *genesis* блок шаље *ordering* сервису у захтеву за придруживање каналу [6]. Трансакција за креирање канала специфицира иницијалну конфигурацију канала и креира се уз помоћ *configtxgen* алата.

4.3 Дефинисање и инсталација паметног уговора

Комплетна пословна логика садржана у паметном уговору се налази у *Hyperledger Fabric SDK* апликацији, написаној у *Java* програмском језику и развијеној уз помоћ *gradle* алата. *Java* омогућава употребу анотација како би се обезбедиле информације о паметном уговору и његовој структури. Класа која садржи кôд паметног уговора анотирана је са *@Contract(...)* анотацијом.

Ова анотација пружа могућност дефинисања додатних информација као што су назив, опис, лиценца и подаци о аутору. Да би класа представљала паметни уговор, поред анотација потребно је и да имплементира *ContractInterface*.

На слици 3 приказана је структура паметног уговора. Методе анотиране са *@Transaction* анотацијом означавају део кода који ће се извршавати као трансакциона функција. Методе које се позивају како би се додала нова трансакција, или како би се изменила постојећа трансакција су анотиране са типом *SUBMIT*, док су остале методе анотиране са типом *EVALUATE*.

Слика 3. *Паметни уговор*

Да би се приступило *ledger*-у, потребно је дефинисати контекст који ће бити доступан свим трансакцијама. Контекст се увек наводи као први параметар трансакционих метода. Да би се приступило стању света, контекст нуди *.getStub()* методу. Објекат који је враћен из ове методе служи како би се формирали упити ка бази података. Систем за пријаву мастер радова садржи једну методу за креирање трансакција односно за додавање нове пријаве на *ledger*. Ова метода садржи две главне провере. Прва провера омогућава јединственост трансакција. Уколико пријава прође ову проверу, позива се метода која имплементира предуслове које студент треба да испуни како би пријава била валидна. Услови да би студент могао успешно да поднесе пријаву су да је прошло најмање 6 месеци од датума уписа студијског програма, и да област из које студент жели да ради мастер рад припада области предмета на студијском програму. Ако су оба критеријума испуњена, креира се нови JSON објекат од параметара који чине пријаву. Креирани објекат се серијализује и додаје у стање света, под одговарајућим кључем. Да би се написани *chaincode* извршавао на чворовима, потребно га је прво исталирати на канал. За инсталацију паметног уговора користи се команда наведена на листингу 1.

```
./network.sh down
./network.sh up createChannel -ca
./network.sh deployCC -ccn basic -ccp ../asset-
transfer-basic/chaincode-java -ccl java
```

Листинг 1. *Покретање blockchain мреже*

4.4 Повезивање апликације са *chaincode*-ом

Да би апликација могла да креира трансакције и проверава стање света на *ledger*-у, потребно је успоставити конекцију са *Hyperledger Fabric SDK*-ом. Прво, апликација пријављује администратора. Креденцијале за администратора генерише *Fabric CA* клијент. Да би се апликација успешно повезала са CA клијентом, потребно је поставити путању до сертификата апликације. Ако су сертификати исправни, *Fabric CA* клијент ће креирати сертификат за администратора и вратиће га апликацији. Сви сертификати се смештају у електрнски новчаник (енгл. *wallet*). Након уписа администратора, потребно је регистровати кориснике. Поступак регистрације корисника је сличан. У трећем кораку, потребно је повезати се на мрежу помоћу *gateway*-а. За успостављање конекције користи се претходно регистрован корисник, односно његови креденцијали учитани из *wallet*-а. Ако је остварена конекција, могуће је добити објекат класе *Network* помоћу којег апликација приступа паметном уговору. Након добијања инстанце паметног уговора, над том

инстанцом је могуће позвати било коју трансакциону методу из класе паметног уговора. На листингу 2 је дат пример повезивања на *blockchain*.

```
try (Gateway gateway = connect()) {
        Network network =
        gateway.getNetwork("mychannel");
        Contract contract =
        network.getContract("basic");
        byte[] result =
contract.submitTransaction("CreateSubmissio
n", String.valueOf(dto.getAssetID()),
        dto.getExpiresAt(),
        dto.getMajor(), dto.getField(),
        dto.getIndex(), dto.getProfesor(),
dto.getProgramName(), dto.getEntryDate());
return new String(result);

} catch (Exception e) {
        System.out.println(e);

}
```

Листинг 2. *Позивање паметног уговора*

4.5 Функционалности система

Апликација прати процес пријаве мастер рада што обухвата попуњавање пријаве, одобравање теме и потом формирање комисије, постављање рада, остављање примедби и на крају одобравање рада. Функционалности се ослањају на позиве ка *blockchain* мрежи а сваки од ових корака обавља једна од улога у систему. Процес пријаве започиње студент, на почетној страници, попуњавањем форме за пријаву. Подаци који обухватају име и презиме, број индекса и студијски програм студента се повлаче из базе података и иницијално попуњавају форму а студент на форми треба да попуни област, студијску групу и наставника код ког жели да пише рад. Када се пошаљу поља форме, креира се *Submission* објекат који се шаље на *blockchain* мрежу.

Позивом одговарајуће методе, активира се код паметног уговора, приказан на листингу 2. Метода ће бити позвана ако постоје кориснички креденцијали у *wallet*-у. Након послате пријаве, систему приступа референт. Референту се на почетној страници се приказују све пристигле пријаве. Референт може да прегледа профил студента где му се прилазује историја полагања испита, просечна оцена и студијски програм за појединачног студента. Када референт прегледа пријаву може да упути захтев наставно научном већу, чиме се мења статус пријаве на *ledger*-у. У следећем кораку, систему приступа неко од чланова наставно научног већа, како би се прегледао пристигли захтев и именовала комисија.

Менор рада може да се пријави на систем како би прегледао листу свих одобрених пријава. Из понуђене листе, ментор може да одабере пријаву за коју ће да дефинише задатак. Студент ком је дефинисан задатак, може да прегледа задатак и да на истој страници постави верзију рада. Постављени рад се хешује помоћу *Bcrypt* алгоритма, након чега се тај хеш шаље на *blockchain* мрежу. Након што студент постави рад, професори који су чланови комисије могу на страници за постављене радове да преузму рад, да оставе примедбу или да га одобре. Ако ни један члан комисије нема примедби на рад, ментор дефинише датум и место одбране, а студент може да провери стање на свом профилу.

5. ЗАКЉУЧАК

С обзиром да се у данашње време све више инсистира на системима који гарантују корисницима безбедност, приватност и поверење, *blockchain* може да се употреби како би се ови захтеви пренели са централизованог тела на технологију, где ће бити довољно ослонити се на исправно функционисање одабраних криптографских примитива и правилној конфигурацији мреже и инсталираних паметних уговора. Овај рад се базирао на употреби *Hyperledger Fabric blockchain* технологије отвореног кода, за решавање проблема аутентичности и поверљивости података који учествују у процесу пријаве мастер рада. Предности система су употреба ECC асиметричног алгоритма за генерисање приватних и јавних кључева, потом генерисање сертификата, као и поседовање комплетне историје свих записа на *ledger*-у. Постојеће решење је могуће проширити на начин који би омогућио различитим универзитетима да приступе мрежи, верификују постојеће радове и дипломе, као и да омогуће другим универзитетима увид и верификацију радова њихових студената. На тај начин, повећава се транспарентност у раду академских заједница, док се истовремено гарантује интегритет података доступних на мрежи.

6. ЛИТЕРАТУРА

[1] *Hyperledger Fabric*, https://wiki.hyperledger.org/display/fabric/HyperledgerFabric (приступљено у септембру 2021)

[2] Christian Gorenflo, Stephen Lee, Lukasz Golab, Srinivasan Keshav, *FastFabric: Scaling hyperledger fabric to 20 000 transactions per second*, 2020

[3] Smart contract, https://hyperledger-fabric.readthedocs.io/en/release-2.2/smartcontract/smartcontract.html (приступљено у септембру 2021)

[4] https://www.docker.com/

[5] *Gossip* протокол, https://hyperledger-fabric.readthedocs.io/en/latest/deploypeer/peerchecklist.html#peer-gossip (приступљено у септембру 2021)

[6] *Planning ordering service*, https://hyperledger-fabric.readthedocs.io/en/latest/deployorderer/ordererplan.html (приступљено у септембру 2021)

Кратка биографија:

Ивана Ковачевић рођена је у Новом Саду 1997. године. Мастер рад на Факултету техничких наука из области Електротехнике и рачунарства – Електрнско пословање одбранила је 2021. год.
контакт: kovacevic.ivana@uns.ac.rs