Институт по Математика и Информатика

Българска Академия на Науките

Секция Математически Основи на Информатиката

Бисер Петров Цветков

Блокчейн технологии и приложението им

Автореферат на дисертация

за присъждане на образователна и научна степен доктор

в професионално направление 4.6. „Информатика и компютърни науки“

Научен ръководител: доц. Христо Костадинов

София, 2023 г.

Увод

Целта на дисертационния труд е да предложи решение за често срещани предизвикателства, свързани с внедряване и поддържане на бизнес софтуерни системи в реална експлоатационна среда. В повечето случаи тези предизвикателства водят до забавяне на реализацията на информационно технологични (ИТ) проекти и до допълнителното им оскъпяване.

Решението на посочените предизвикателства е реализирано посредством блокчейн(англ. blockchain)[1-10] базирана архитектура, която удовлетворява изискванията на съвременния софтуер за доставка и внедряване на бизнес софтуерни системи. То обхваща етапите от жизнения цикъл на разработка на системите (SDLC)[11-14], като специално внимание е обърнато на фазата, свързана с внедряването на софтуера при крайните клиенти, както и последващи актуализации на версиите на софтуера. Важността на този процес изисква той да става по надежден начин, с минимално прекъсване на бизнес процесите и пълна проследимост на отговорностите на всеки един участник. В архитектурата на блокчейн базираната SDLC система е предвидено, чрез умни договори да се автоматизират стъпки, свързани с конкретни отговорности и ключови показатели за ефективност (англ. key performance indicators - KPI), така че своевременно да се уведомят засегнатите страни. Съществено място заема и анализа на сигурността и надеждността на SDLC процесите, гарантирана от блокчейн протоколи, основаващи се на криптографски алгоритми.

В дисертацията се прави и исторически преглед от възникване на Биткойн (англ. Bitcoin)[15-18] криптовалутата до появата на модерни блокчейн платформи, поддържащи умни договори.

Важна част от дисертацията е разработването на прототип, който демонстрира как основните модули, връзките между тях и комуникационните протоколи решават поставените цели спрямо зададената архитектура. Прототипът е изграден върху EOSIO[19-21] блокчейн платформата, която е с отворен код и се отличава с обработка на трансакциите по бърз, мащабируем и сигурен начин. За хранилище за данни с голям размер се използва InterPlanetary File System (IPFS)[22-24] - ново поколение децентрализирана разпределена мрежа за съхранение, наречена междупланетна файлова система.

Структура на дисертацията

Дисертационният труд е разделен на увод и пет глави.

В първа глава е направен обзор върху DLT технологиите, техни ключови характеристики, видове и класификации.

В глава 2 е представена традиционната SDLC област с нейните ключови сценарии, участници в процеси, както и предизвикателства при внедряване и актуализиране на софтуера. Предизвикателствата са описани и в посока изисквания към новата SDLC система, която по иновативен начин да преодолее трудностите, свързани с типичните стъпки за обновяване на софтуера със специален фокус върху сигурността на системата, както и с отговорностите на всеки един участник.

В глава 3 е обоснован изборът на блокчейн платформа и технологични средства за реализиране на SDLC системата. Описани са характеристики на различните съществуващи платформи и анализ на това как техните специфики биха могли да адресират предизвикателствата в SDLC областта. След като избора на платформа е направен, а именно EOSIO, се навлиза в повече детайли на спецификите и функционирането на тази платформа. Най-съществената част, представена в глава 3, е дефинирането на иновативния SDLC блокчейн дизайн и анализ на това как той преодолява съществуващите класически SDLC предизвикателства.

В глава 4 е описан прототип, базиран на иновативната архитектура, като детайлно е описана средата, в която той работи.

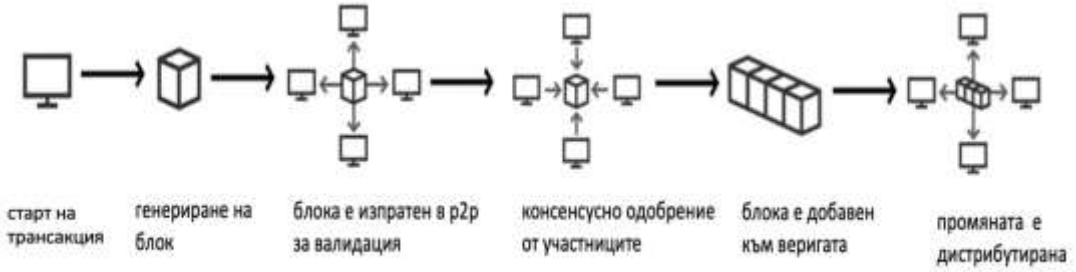
В заключителната глава 5 са обобщени резултатите от разработката на новата SDLC система и са дадени насоки за бъдещо и развитие. Описани са приносите на дисертационния труд и научните публикации по дисертацията.

Глава 1. Обзор и анализ на технологията на разпределените регистри (DLT)

Блокчейн е най-разпространения вид технология на разпределените регистри (distributed ledger technology, DLT), която се състои от нарастващи списъци със записи, наречени блокове, които са сигурно свързани заедно с помощта на криптография. Всеки

блок съдържа криптографски хеш на предишния блок, клеймо за време и данни за трансакция. Времевият печат доказва, че данните за трансакцията са съществували, когато блокът е бил създаден. Понеже всеки блок съдържа информация за предишния блок, те ефективно формират верига, като всеки блок се свързва с тези преди него. Трансакциите в блокчейн са необратими, понеже след като бъдат записани, данните във всеки един блок не могат да бъдат променени със задна дата, без да се променят всички следващи блокове.

Блоковата верига се съхранява в мрежа от разпределен тип, където всеки участник в мрежата разполага с копие на веригата в компютъра си. По този начин няма точно едно определено главно копие, както и няма риск от срив, загуба или манипулиране на информация. Участниците в мрежата са равноправни (*peer to peer*) и спазват определен протокол за валидиране на новите блокове. Веднъж валидиран и записан нито един блок не може да бъде променян без одобрение (консенсус) на останалите участници в блокчейн веригата.

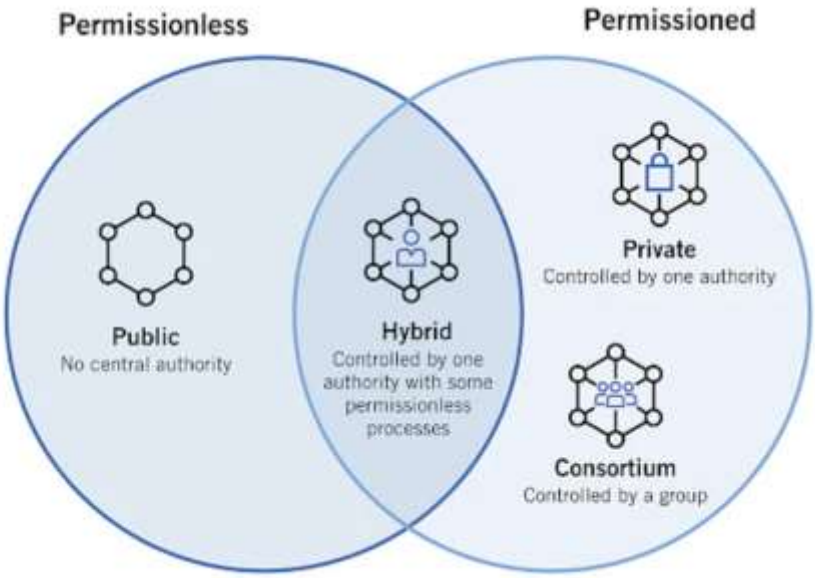


В процедурата по промяна или добавяне на данните в регистъра се използват различни методи за одобрение. Начинът на взимане на решение за достоверност се нарича консенсус между участниците. Това е процес на одобрение, потвърждаване на данните от всички или от мнозинството от участниците и завършва с подписване на трансакциите. В биткойн мрежата например се използва протоколът „Доказателство за работа” (PoW). В зависимост от това как си комуникират възлите в мрежата, както и дали мрежата е част от дадена корпорация или е публична, съществуват различни консенсусни протоколи [25-28].

Видове блокчейн

Блокчейн платформите могат да бъдат изградени по различен начин, с различни права на възлите в мрежата, с различен достъп до мрежата, както и с различни

механизми за консенсус. В зависимост от това може да се дефинират следните видове блокчейн платформи.



Публичен Блокчейн

Публичният блокчейн е децентрализирана мрежа, към която всеки може да се присъедини и да участва в нея. Публичните блокови вериги позволяват на всички възли да имат равен достъп до блоковата верига, дават възможност за добавяне на нови блокове от данни и за валидиране на съществуващи блокове от данни.

Частен Блокчейн

Частните блокчейн са централизирани и управлявани от лице или организация, която решава кой може да има достъп до блокчейна и да бъде добавен като възел. Трансакциите в частната блок верига не са публично видими, но се проверяват чрез процес на консенсус между членовете на мрежата.

Хибриден блокчейн

Хибридите блокчейни съчетават характеристиките на публични и частни блокчейн мрежи. Този тип блок верига често се използва в бизнес приложения, при които множество организации трябва да споделят сигурно данни. Хибридната блокчейн има способността да споделя публично определена информация, като същевременно запазва друга информация поверителна. Това позволява по-голяма сигурност и прозрачност, като едновременно с това се запазва известна степен на поверителност.

Консорциум блокчейн

Група от компании или организации управляват процес на даване на разрешение за достъп до блокчейн. Те са по-децентрализирани от частния блокчейн, което дава по-голяма сигурност. Предварително определени възли управляват консенсусните процедури в блокчейн на консорциум. Този вид блокчейн има валидиращ възел, чиято основна функция е да извършва иницииране на трансакция, получаване и валидиране.

Предимства и ключови характеристики

Благодарение на използването на p2p мрежи, криптографски функции и консенсусни алгоритми DLT предоставя следните предимства:

- Децентрализация - информацията е винаги налична, понеже всеки компютър в мрежата разполага с копие на регистъра;
- Прозрачност - информацията за трансакциите се съхранява в публичното пространство. Тези данни обаче не могат да бъдат променяни;
- Неограниченост - теоретично регистърът може да бъде допълнен с вписвания до безкрайност;
- Достъпност – всеки има достъп и може да записва, естествено това е валидно за дадени типове DLT;
- Надеждност – проверка на информацията чрез консенсус, както и невъзможност за промяна в нито един момент след записване в регистъра;
- Сигурност – използват се надеждни криптографски механизми за валидиране и защита на информацията;
- Информацията може да е анонимна или поименна – отново се дължи на използвани криптографски функции (публичен-частен ключ);
- Автоматизация – благодарение на умните договори (smart contracts) може да се направи автоматизация на изпълнението;
- Ефективност – поддържането на регистъра и превода на трансакции не изисква високи вложения;

- Бързодействие – трансакциите се случват мигновено в зависимост от избора на DLT.

Всички тези качества на DLT могат драстично да променят текущите бизнес процеси, като ги направят по ефективни, достъпни и надеждни.

Приложимост на DLT в бизнеса

Благодарение на ключовите си характеристики блокчейн приложения имат потенциал и се ползват от много индустриални сектори[29-39]. Те осигуряват по-добро качество на продуктите, по-голяма удовлетвореност на потребителите, намаляват разходите, увеличат прозрачността и справедливостта и подобряват ефективността на пазара. Блокчейн позволява ефективно съхраняване на данни за трансакции, клиенти и доставчици в прозрачна, непроменяема онлайн книга.

Блокчейн успешно се прилага на практика в следните сценарии:

- Вериги за доставки и логистика
- Здравеопазване
- Търговия на дребно и електронна търговия
- Финанси
- Недвижими имоти
- Медия
- *NFT* пазари
- Тежка промишленост и производство
- Музика
- Трансгранични плащания
- Интернет на нещата
- Игри
- Сигурност на личните данни

- Правителство и гласуване
- Реклама
- Създаване на оригинално съдържание
- Автомобилна индустрия
- Интелигентни договори

Глава 2. Предизвикателства пред съществуващите системи за управление на жизнения цикъл на софтуерни проекти

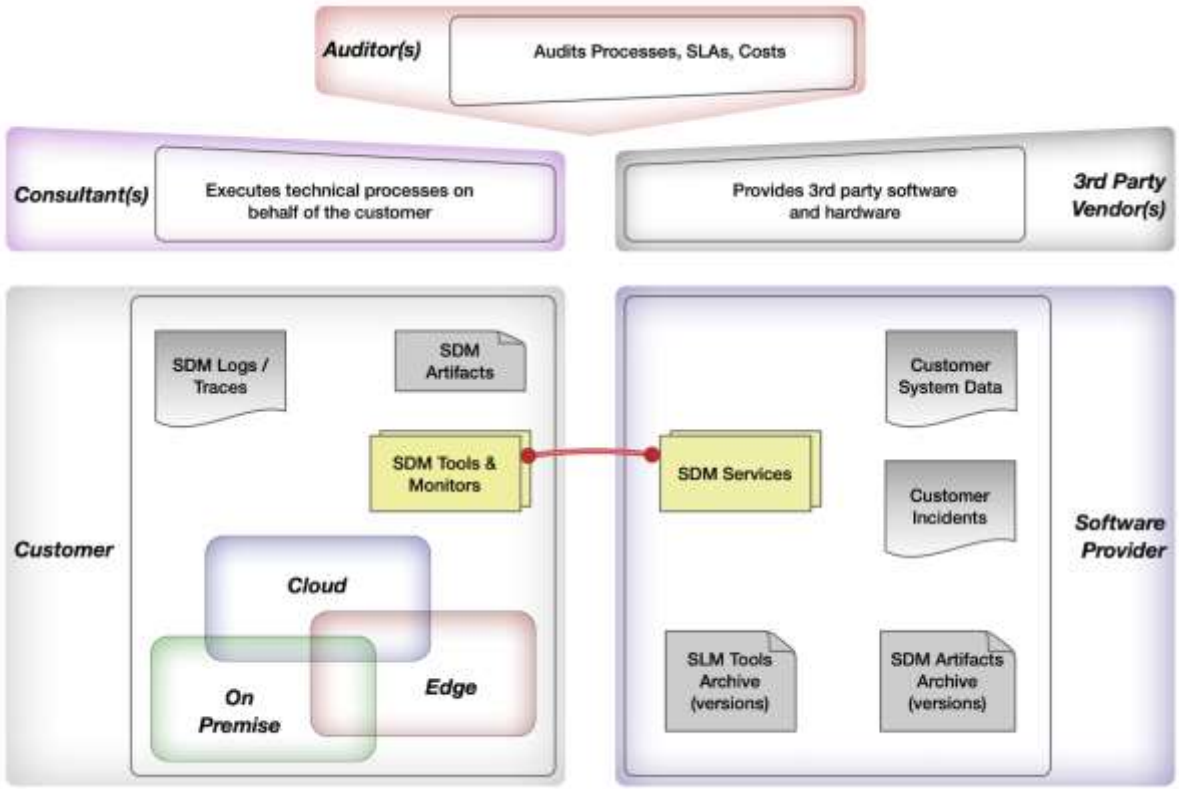
Докато в момента усилията на много разработчици са насочени към преобразяването на класическия бизнес, има една ИТ област, която също би могла да се възползва от блокчейн, а именно управлението на жизнения цикъл на разработка на софтуерни системи(systems development life cycle - SDLC).

При SDLC голяма част от предизвикателствата настъпват във фазата на внедряване, конфигуриране, обновяване и поддръжка на софтуера при крайния клиент(System Deployment and Maintenance - SDM). Точно в тази фаза използването на блокчейн би решило типичните проблеми, съпътстващи първоначалното инсталиране и конфигуриране на софтуера в продуктивна среда.

Типични SDM сценарии

Един ИТ сценарий, който е близък до реална бизнес система се състои от различни софтуерни продукти, които работят на локални сървъри, в облачна среда и интегрирани IoT устройства. Типичните участници, които са отговорни за внедряването на една такава сложна система са:

Доставчикът(Software Provider) на софтуера е ИТ компанията, която е разработила софтуера и предлага постоянна поддръжка за своите клиенти. Доставчикът доставя софтуерни артефакти за процедурите на SDM, които са инсталационни файлове, системни надстройки, актуализации, *пачове*(поправки на грешки), инструменти за конфигуриране и други. Отговорностите на доставчика на софтуер включват доставяне и поддръжка за техните продукти по дефинирани строги индикатори за време и качество (KPI) за справяне с различни потенциални клиентски проблеми.



Проекти с външно или публично финансиране обикновено имат определен **одитен орган(auditor)**, назначен за проекта. Въпреки че одиторите не допринасят пряко за техническото изпълнение, те са важна част от цялостния проект и се нуждаят от

надеждна информация за състоянието и ресурсите, както и за предстоящи и спазени срокове. Одиторите трябва да преценят дали проекта се реализира с договорената скорост и в дадения бюджет.

Техническите консултанти имат ключов принос за реализирани проекти особено в частта на внедряване. Те изпълняват технически стъпки от името на клиента. Тяхната област на действие обикновено е локалната среда на клиента, компоненти и услуги, работещи в облака, както и крайните IoT устройства. Консултантите също комуникират на техническо ниво с всички други участващи страни. Много често техническите консултанти са сертифицирани от доставчика на софтуер и са достатъчно добре подготвени за използваните SDM процедури.

В повечето случаи един софтуерен **доставчик** не може да осигури на абсолютно всички компоненти, участващи в изграждането на крайната система. Много често хардуера, както и операционните системи са произведени от други компании, които обобщено се назовават като **доставчик трета страна(3rd party vendor).** Така различните производители на операционната система, базите данни, сървърите и IoT хардуер добавят към сложността на SDN процедурите. Тестването за съвместимост на хардуера и софтуера, доставен от трети страни, е съществена част от всяка SDM процедура.

Всичко това се прави в полза на **крайния клиент(customer)**, който заплаща и използва предоставения софтуер. Клиентите са участниците, които използват продуктивно доставения софтуер и участват в дефинирането на KPI за SDM процедурите на поддържаната система. Максималния период за недостъпност, изискванията за наличност, изискванията за защита (копия) на информация в случай на бедствие, достъпът до фирмените ресурси и други KPIs се съгласуват с клиента.

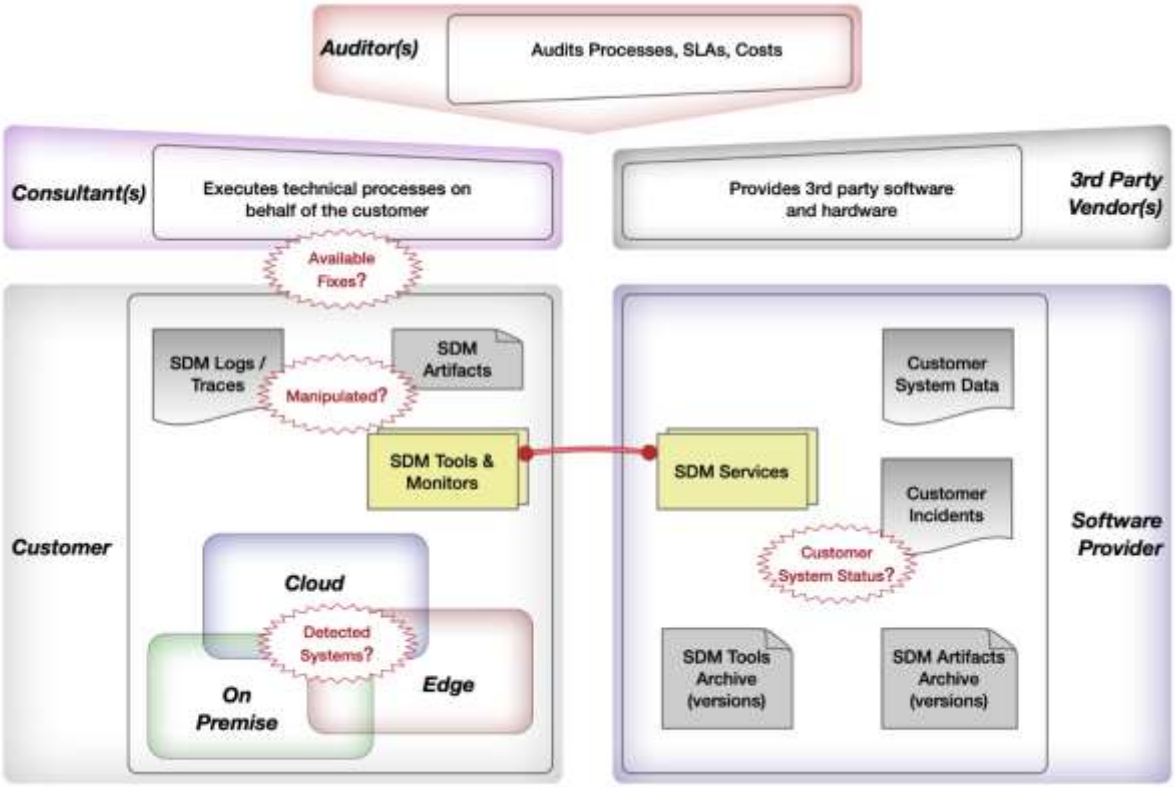
В първия етап всички участници са ангажирани с началното инсталиране на софтуер. Клиентът заплаща софтуера. Техническият консултант сваля инсталационните файлове и започва самата инсталация. По време на инсталацията всички важни стъпки се записват в така наречените лог файлове. След успешна инсталация следва процес на конфигурация, включваща настройки и връзка към бази данни, конфигуриране на адресите на модулите, връзката между тях и други настройки.

Друг сценарий, който идва в даден период след първоначалната инсталация, е обновяване на версията на софтуера. При него обикновено системата спира за определен

интервал от време, всеки модул се подменя с по-новата му версия и в някои случаи таблиците в базите данни се обновяват. По времето на обновлението отново всички важни стъпки се записват в лог файлове.

Разновидност на обновлението е така наречената миграция. При нея разликите между старта и новата версия са толкова големи, че единствения вариант за преход е новата версия да се инсталира на чисто. След което се мигрират само данните от старата система към новата, в доста случаи като се прилагат трансформационни правила.

Предизвикателства с традиционния SDM



Основните предизвикателства пред SDM са спазване на крайни срокове, ненадвишаване на договорения бюджет и коректно функциониращa система. Въпросът за дефиниране и проследяване на отговорностите на всеки един участник във всеки един етап е от съществено значение.

Тези предизвикателства са трудно преодолими в текущо използваните SDM процедури понеже взаимоотношения между участниците във фазата на инсталация, интеграция и обновление са комплексни, а нивото на доверие между тях, в повечето

случаи, не е изградено. В зависимост от проблемната област се разглеждат следните категории предизвикателства:

1. **Цялост и сигурност на системата**

След начална инсталация или обновление често се случва някои артефакти на SDM – изпълними файлове, скриптове за конфигурация или лог файлове - да се манипулират в мрежата на клиента. Понякога изтриването на такива файлове дори е част от установените процедури на клиентската среда. В някои случаи лицето, което има достъп до скриптовете, може съзнателно да избере да не прилага (случайно или умишлено) определена корекция или корекции към системата. По този начин системата може да бъде оставена в неизправно състояние или нейната сигурност да бъде компрометирана. Ако лицето има достъп до регистрационните файлове и логовете, то може да прикрие всички свои следи и да скрие подобни манипулации от клиента и другите страни.

По-късно в случай на спор доставчикът на софтуер може да бъде държан отговорен за всички щети, причинени от използването на компрометираната система, без той да е наясно какво се е случило. В тези случаи също е много сложно за одиторската фирма да прецени кой е отговорен за даденото състояние на системата и кой дължи неустойки. Но наред с отговорността, по-големият проблем е, че целостта на системата е нарушена за дълъг период от време и трудно се намира начин тя да бъде възстановена.

2. **Видимост на данните**

Видимостта на данните в средата на клиента е важен аспект от поддръжката на всяка система. Доставчикът на софтуер трябва да знае какви приложения се използват от клиентите му, на каква версия са те и дали са правилно обновени и конфигурирани. От друга страна, клиентът и неговите консултанти имат нужда да знаят за всички нови версии на предоставения софтуер. Всички корекции на сигурността трябва да бъдат приложени възможно най-скоро, за предпочитане преди да станат публично достояние и преди да се извършат успешни атаки.

Също така е важно и откъде се взима байт кода на инсталационните файлове, подписан ли е от доставчика и дали хеш кода отговаря на оригиналния. Всички тези теми, свързани с видимостта, са изключително важни за добрата поддръжка на продуктивните системи.

3. Времеви интервал на недостъпност на системата и управление на риска

За много SDM процедури е важно да се предскаже очакваната продължителност на стъпките на процедурите, както и натоварването на ресурсите. Един от най-съществените етапи за прогнозиране е интервалът на прекъсване на системите, в които те няма да са налични. Освен това някои от процедурите може да съдържат критични стъпки, които потенциално биха могли да причинят допълнителни проблеми в зависимост от типа използвани бази данни, операционна система и други специфики на средата. Затова е добре информация от процедурите, които са се е изпълнили при клиенти, да се съхранява (анонимно) исторически и на базата на тази информация да се прави оценка на риска. Имайки тази информация доставчикът на софтуер може да прецени риска от прилагане на процедура при нов клиент или да адаптира някои конфигурационни параметри. Например, ако базите данни са на даден производител, а операционната система на друг производител, то биха могли да се приложат конкретни оптимални параметри за подобна среда. Друг пример е, ако се знае че при даден размер на базата данни процедурата за обновяване на софтуера отнема 2 часа, а при по-голям размер отнема 8 часа, то много прецизно клиентът може да прецени кога да приложи обновлението, така че времето на недостъпност на системата да се отрази на възможно най-малък брои негови крайни потребители.

4. Прогнозиране на поведението на системата

Понякога забавянето на реакция от страна на бази данни, или възникване на предупредителни съобщения в системата, които се записват в лог файловете, са ранна индикация, че системата е под риск и е вероятно да се срине. Клиентите и техническите консултанти не обръщат внимание на тези ранни индикации или не могат правилно да ги разчетат. Ето защо би било добре софтуерният доставчик да получава достъп до тази информация и да може да предотврати срив при конкретен клиент, но и още по-важно да анализира, прецени и предостави подобрена версия на всичките си клиенти, които използват подобна среда. Често се наблюдават случаи на критично поведение на системата причинени от прекалено, непланирано натоварване, което не съответства на начално заявеното(хардуерът и изчислителната мощ не отговарят на начално заявената). Но дори в този „лесен“ случай крайният клиент в повечето случаи е затруднен да стигне до основната причина. Ето защо ключови параметри като натоварване на процесор, обем

на база данни, скорост на изпълнение е добре да бъдат конфиденциално споделени с производителя на софтуера.

5. **Отговорности и SLA**

Начина на договаряне на срокове за доставка на компонентите за продуктивна система, било то хардуер, продуктов софтуер , операционна система или бази данни все още на моменти се осъществява в устни договорки, по телефон или емайл. Също така последователността кой участник какво трябва да свърши, след кого и до коя дата е много трудно да се проследи.

Проследимостта в повечето случаи е свързана с преглед на официални протоколи от срещи и проследяване коя среща е била последна със съответно последни договорки. Тази размитост на отговорности е причина за забавяне на проекти, както и за това одиторските фирми да са максимално затруднени при кого е отговорността за конкретно забавяне. Съответно затруднено или понякога дори невъзможно е да се търсят компенсации за забавени проекти.

В глава 3 е описана иновативна блокчейн архитектура, която адресира тези 5 предизвикателства.

Глава 3. Архитектура на DLT базирана SDM система

В тази глава е описана иновативна блокчейн базирана архитектура, която цели да реши проблемите които съществуват по време на внедряването и поддръжката на софтуерни системи в комплексна продуктивна среда.

Избор на DLT платформа

За по-прецизен избор на най-подходяща платформа, решаваща SDM предизвикателствата, ще се направи детайлно сравнение на два публични блокчейна: Ethereum[40-42] и EOS и два корпоративни: Hyperledger Fabric[43-45] и R3 Corda[46-48].

Един от основните критерии за избор какъв вид DLT да се използва е публичен или корпоративен. Разходите, свързани с използването на публичен DLT и разпределено съхранение, също варират значително. Корпоративните DLT са по-гъвкави за разработване – техните интелигентни договори обикновено са променливи – с

възможност за надграждане и разширяване. Но когато се използва корпоративно DLT, трябва да се вземе под внимание сложността за поддръжката на самата DLT.

Ethereum: Ethereum е първата блокчейн платформа, поддържаща интелигентни договори. Тя използва консенсусен механизъм за доказателство за дял (PoS). Времето за създаване на блок е средно 15 секунди, а общата производителност е около 15 трансакции в секунда в световен мащаб. Ethereum съдържа абстрактен слой с вграден пълен език за програмиране на Turing, позволяващ всеки да пише интелигентни договори и децентрализирани приложения, където може да създава свои собствени произволни правила, формати на транзакции и функции за преход на състоянието. Езикът на интелигентния договор е Solidity – клонинг на Java скрипт, проектиран специално за изпълнение на интелигентни договори на Ethereum. Отличителна черта на Ethereum е, че неговите интелигентни договори са неизменни.

EOS: EOS е трето поколение блокчейн платформа, базирана на делегирано доказателство за дял (DPoS). В DPoS всеки притежател на EOS може да гласува за производители на блокове, на които има доверие. Избрани са общо 21 производители които отговарят за обработката на трансакциите, като ги хешират в блокове. Блокове се произвеждат точно на всеки 0,5 секунди и точно един производител е упълномощен да произвежда блок във всеки даден момент.

Интелигентните договори на EOS обикновено са написани на C++. По подразбиране интелигентните договори на EOS са промениими – поправките и разширенията се извършват сравнително лесно. В случай, че разработчикът реши да превърне съществуващия интелигентен договор в неизменен, има опция за премахване на публичните ключове на неговия акаунт. Това прави договора практически неизменим, тъй като не може да се използва частен ключ за внедряване на промени.

Наред с ценовото предимство, EOS предоставя и много добра среда за разработване на децентрализирани приложения.

Hyperledger Fabric: Hyperledger Fabric е част от общността с отворен код на Hyperledger. Той е предоставен от IBM и е създаден специално за корпоративна употреба. Не разполага със собствена криптовалута. Интелигентните договори на Fabric се наричат верижен код и обикновено се разработват на Golang. Верижният код на Fabric е единственият начин за достъп или модифициране на данни в Hyperledger Fabric.

R3 Corda: Corda е DLT, което не е блокчейн, създаден от R3 Consortium за основно използване във финансови институции. Той има подобрена мащабируемост в сравнение с всеки класически блокчейн поради иновативната си архитектура. За разлика от Hyperledger Fabric, консенсусът за Corda се постига на ниво трансакция (няма блокове) и винаги включва взаимодействие с един или повече възли на така наречения нотариален клъстер, за да се гарантира уникалността на трансакцията. Интелигентните договори на Corda обикновено са написани на езика Kotlin.

Сравнение

Крайните потребители взимат решение за използване на дадена системата въз основа на ползата, която им носи нейната използваемост, първоначалните разходи и разходите за поддържка. Други водещи характеристики за избор на платформа са функционалност, производителност, надеждност, сигурност, мащабируемост. Не на последно място от значение е и големината на екосистемата, която ползва платформата, дали тя постоянно се подобрява, дали се пишат много нови функционалности, както и дали средата за писане на нови приложения е лесна и интуитивна за работа.

Сравнението между дадените четири DLT подчертава разликите, които вече съществуват между различните технологии. Използването на корпоративни DLT (HL Fabric и R3 Corda) изисква значителни разходи за инфраструктура, която да бъде създадена и поддържана от участващите страни. Това включва разпределение на ресурси, както за хардуер и за софтуер, така и персонал, отговорен за актуализации и промени в конфигурацията на системата. За организационно разпределена система тези разходи могат да бъдат значителни, ако се извършват по начин, който не разчита на доверие към други участници в процеса. Като допълнителен фактор трябва да се има предвид ,че ако вече има добро ниво на доверие между всички страни, може значително да се опрости цялостната архитектура, като се използват класически бази данни вместо DLT.

Използването на публични DLT (Ethereum и EOSIO) от друга страна гарантира доверието между всички участници понеже е базирано на публична p2p мрежа. Инфраструктурата на публичните DLT вече е създадена и се поддържа от други страни, което дава ценово предимство и улеснена поддръжка. DApps, работещи на публичен DLT, могат да се консумират директно дори от мобилни устройства. Тези свойства на

публичните DLT им дават решително предимство пред корпоративните DLT, когато се отнасят за обслужване на разпределена SDM система.

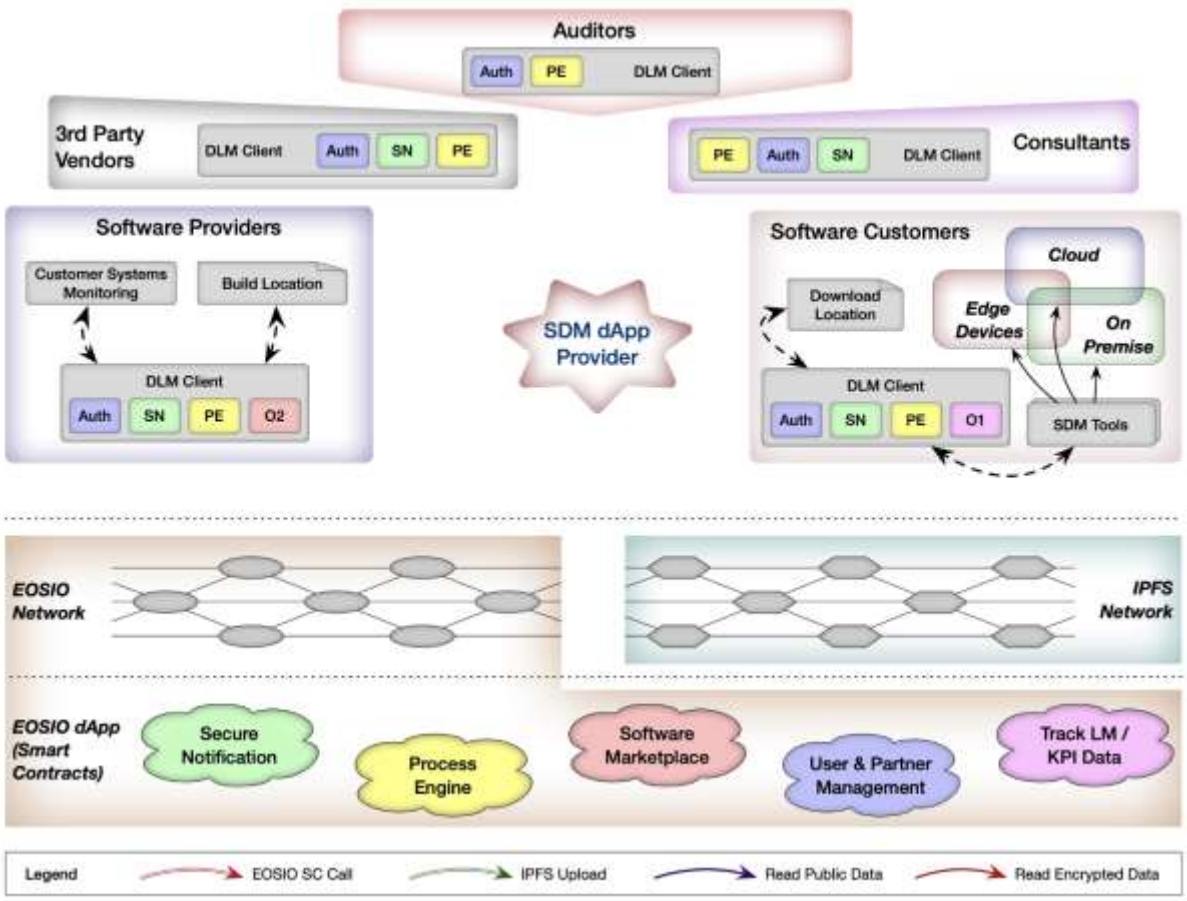
Изброените прилики и разлики, както и ключови характеристики са обобщени в следната таблица.

	Ethereum	EOS (Mainnet)	HL Fabric	R3 Corda
Type of DLT	Public	Public	Permisioned	Permisioned
Consensus Type	Proof of Work	Delegated Proof of Stake	Configurable per channel	Notary service + UTXO
Smart Contract Type	Immutable	Mutable/Immutable/BP	Mutable	Mutable
Smart Contract Programming Language	Solidity	C++	Golang	Kotlin
Legal Binding Agreement	- - -	Ricardian Contracts	- - -	Ricardian Contracts
Deployment Cost for dApp Publisher/Operator	Gas price for deployment	Cost: RAM (code + data)	(Distributed) Infrastructure	(Distributed) Infrastructure
Maintenance Cost for dApp Publisher/Operator	- - -	Cost: RAM delta (+/-)	Maintain (Distributed) Infrastructure	Maintain (Distributed) Infrastructure
Motivation to run DLT	Miner's reward per block (inflation)	Block producers' reward (inflation)	Infrastructure maintained by involved parties	Infrastructure maintained by involved parties
End user joining	Free	~ Free minimum RAM, CPU, NET	Restricted	Restricted
End user costs	Gas cost per dApp call	Practically free / Replenishable resources	- - -	- - -
On-chain data visibility	All persistent dApp data is visible globally	All persistent dApp data is visible globally	Full visibility per Fabric channel	Only participants see transactions. Transaction proof may be shared.
Scalability	Limited due to PoW used	Improved due to DPoS and limited number of BPs	Each channel is separate blockchain	Excelent, Native sharding support
Consensuis	Proof of Work	Delegated Proof of Stake	Configurable per channel	Verify: Required signers Unique: Notary service
Security	Top 3 Mining pools control 64% of hashrate	Top 21 Block Producers votable each 63 sec.	Customizable per scenario	Notary service
Transactions per Second	About 15 transactions per second globally	Limited to 4000 TPS for Mainnet	Configurable per channel	N/A
Block producing time	~ 15 seconds	0.5 seconds	Configurable per channel	N/A

Ethereum официално преминава към механизъм за консенсус Proof of Stake (PoS) през 2022 г. като по-сигурен и енергийно ефективен начин за валидиране на трансакции и добавяне на нови блокове към блокчейна. Базирано на направения анализ, взимайки предвид всички важни характеристики, EOSIO е по-подходящ избор на платформа за разработката на SDM прототип. Наред със скоростта, доверието идващо от публичния му тип, както и популярния език (C++) използван за писане на умни договори има и още един водещ фактор за избора на EOSIO, а именно удобната среда за писане на dApp приложения.

Архитектура на SDM системата, базирана на DLT

Основни компоненти на SDM архитектурата са публичен EOSIO блокчейн, разпределена система за съхранение на файлове IPFS, единичен dApp, който е публикуван в EOSIO и използва указатели към файлове съхранявани в IPFS. Всеки от участниците в разпределените SLM процеси използва един или повече локално инсталирани Distributed Lifecycle Management (DLM) *клиенти* за комуникация с блокчейна/dApp-а, като тази комуникация в повечето случай ще стигне и до другите участници в SDM процеса и/или до базата данни (IPFS-а).



В основата на архитектурата са пет умни договора [49-50], които може да се дефинират като модули: *модул за управление на потребителите* (User and Partner Management), *модул за управление на процеси* (Process Engine), *модул осигуряващ сигурна комуникация* (Secure Notification), *модул за софтуерна потребителска кошница* (Software Marketplace) и *модул за проследяване*(Track LM/KPI Data) съдържащ данни от

клиента на LM/KPI данни. Тези модули са част от dApp-а и ще бъдат разгледани в детайли. Характерното за тях е, че те са моделирани да следват важните етапи от внедряването и поддръжката на софтуерни системи и през тях се осъществява четенето и писането в блокчейн.

Ролята на локалните разпределени *клиенти* (DLM) е да осигури възможност на потребителите да се удостоверяват и да използват интелигентните договори на SDM dApp, както и да закачат ресурси под формата на файлове в IPFS. Данните, които DLM записват в EOSIO или IPFS най-често ще бъдат криптирани чрез публичните ключове на страните от процеса, които имат право да четат данните. DLM може да изпълнява автоматично различни стъпки от SDM процесите. Това е особено важно за някой видове блокчейн *oracles*, които регулярно публикуват данни в SDM dApp. При тях се имплементира модул за стартиране на процеси (scheduler), който изпълнява стъпки от процеса на зададени интервали. Например един *oracle*, който прикачва към SDM процеса логове или статус на работата на конкретни клиентски системи, които са релевантни за дефинираните KPI, може да се задейства веднъж дневно или при настъпването на конкретни изключителни събития.

Ролята на IPFS е да осигури база данни за големи файлове и е важна за SDM процеса. Данните съхранявани в EOSIO са сравнително скъпи за съхранение, сравнени с IPFS. При наличие на големи количества данни е много по-ефикасно те да се съхраняват в IPFS, отколкото да използват ресурсите на EOSIO CPU и EOSIO Net докато се качват данните в EOSIO и освен това блокират EOSIO RAM, докато се съхраняват. Разликата при цената за съхраняване на данни в EOSIO и IPFS е от няколко порядъка.

Указателите към файлове в IPFS са хешовете им, които се съхраняват в dApp-а на EOSIO. По този начин в системата се гарантира непроменимостта на IPFS данните, използвани от всеки SDM процес.

Много често съхраняваните в IPFS данни трябва да са достъпни само за определена група участници в SLM процесите. Пример за такива файлове са логове от изпълнението на работата на системите при клиентите или двоични и други файлове, свързани с критични фиксове, отнасящи се до подобряването на сигурността на дадени компоненти, които все още не са публично достъпни. Защитата на такива данни в IPFS се извършва чрез криптиране на файловете с генериран индивидуален за файла симетричен ключ, като самият ключ се криптира чрез публичния ключ на конкретните участници в процеса

и се записват в EOSIO dApp. По този начин единствено участникът, чийто публичен ключ е използван, може да декриптира симетричния ключ и да го използва, за да декриптира съдържанието на конкретния файл във IPFS.

Oracles[51] предоставят външни данни за интелигентни договори, които работят на блокчейн технология. Те по същество са форма на комуникация между външния свят и света на блокчейн. Пример е oracle O1, който докладва чрез DLM клиента различни KPI на системите към платформата за интелигентни договори на EOSIO и IPFS. Друг пример е O2, който отговаря за всички необходими взаимодействия с блокчейна по отношение на каталога на софтуерни продукти.

Адресиране на изискванията към съвременна SLM система

Описаните в глава 2 изисквания към съвременната SLM система, а именно *„Цялост и сигурност на системата“, „Видимост на данните“ , „Времеви интервал на недостъпност на системата и управление на риска“, „Прогнозиране на поведението на системата“, „Отговорности и SLA“* са адресирани в предложената архитектура.

Цялостност и сигурност на системата

Важен аспект за работата на съвременните софтуерни системи е гаранцията, че кодът, който се изпълнява на клиентските системи, е същият код, който софтуерният доставчик е дал на клиента. В случая на блокчейн архитектура DLM *клиентът* ще провери през модул *Software Marketplace* коя е последната версия на дадения продукт, както и кой е хеша му в IPFS. След което може да свали локално необходимите файлове, като отново сравни техния хеш.

Видимост на данните

Друг от обичайните проблеми на SLM процесите е недостъпността на данните за процеса, събирани в реално време или след изпълнението му. С предложената разпределена SDM система това е адресирано, като всички релевантни ресурси при подготовката на процеса, както и при изпълнението му мога да бъдат споделени с всички или избрана група от участниците в процеса. В този случай EOSIO и IPFS изпълняват ролята на доверена среда за разпространение и съхранение на информацията за конкретната инстанция на SDM процеса.

Времеви интервал на недостъпност на системата и управление на риска

Един от най-важните за клиента параметри на SLM процесите са риска за системата и времето, когато системата няма да бъде налична за използване. Рисковете при SLM процесите са от различно естество – надвишаване на планираното време на недостъпност на системите, надвишаване на планирания бюджет, непълно функционираща система и в най-лошите случаи – неуспешно завършила SDM процедура или дори загуба на данни. Събирането на анонимизирани данни от изпълнението на различни SLM процедури при клиенти, в доверена блокчейн среда, спомага за създаване на реалистична оценка за рисковете от бъдеща SLM процедура.

Прогнозиране на поведението на системата

Понякога забавянето на реакция от страна на бази данни или възникване на предупредителни съобщения в системата, които се записват в лог файловете, са ранна индикация, че системата е под риск и е вероятно да се срине. Много често клиентът и техническите консултанти не обръщат внимание на тези ранни индикации или не могат правилно да ги разчетат. Чрез модула *Track LM/KPI Data* тази информация ще достига до софтуерния доставчик и така проактивно може да се предотвратяват критични ситуации.

Отговорности и SLA

SDM процедурите често са сложни процеси, които изискват приноса на голям брой участници. Тези участници обикновено са финансово-зависими от резултата на процедурата и не са непременно организирани в йерархична структура. Поради това отговорностите за изпълнението стъпките, особено на такива, които водят до настъпване на събития, свързани с финансови, репутационни и други щети, понякога е трудно да бъдат еднозначно определени.

Предложената система позволява процесът да се следи от много участници, като всяка стъпка от него да има еднозначно определен отговорен участник. По този начин изпълнението на всеки отделен етап от целия процес може да се проследи по време и качество на изпълнение от останалите участници.

От изложеното до тук е видно, че прозрачността, целостта, сигурността и анализа на системите е гарантиран по най-надежден начин, базиран на доказани криптографски алгоритми и с осигурен достъп в разпределени p2p мрежи.

Основни модули и канали за комуникация

Следните интелигентни договори на предложеното SDM приложение са в основата на dApp-а: договор за управление на потребителите (User and Partner Management), двигател на процеси (Process Engine), интелигентен договор, осигуряващ сигурна комуникация (Secure Notification), софтуерен пазар (Software Marketplace) и модул, съдържащ данни от клиента за проследяване на KPI данни (Track LM/KPI Data).

Интелигентният договор за управление на **потребителите** следи връзката между различни страни, роли и процеси. Използва се за валидиране на оторизация за EOSIO акаунти, както и за да задейства определени промени в състоянието на процесите. Една от основните му цели е да поддържа модел за всеки процес, кои са участниците в него и какви роли изпълняват. В някой от ролите е допустимо да има повече от един участник. Например в един SDM процес може да има няколко доставчика на софтуер, консултанти, доставчици на трети страни, както и одитори. Може да се смята, че клиентите обикновено са уникални за всеки процес, но дори тази интерпретация може да има варианти в ситуации, когато група участници имат общи изисквания и нямат директна йерархична връзка помежду си. В такива случаи един SDM процес може да има няколко участника, дефинирани като клиенти, но трябва да се избере кой от участниците е отговорен за всяка отделна стъпка, когато се изисква действие на клиента. За всеки участник системата пази данни за използвания публичен транспортен ключ, който да се използва от модула за сигурна комуникация. Данните за процесите и взаимосвързаните участници се попълват от доставчика на софтуера или от клиента, според вида на конкретния процес. Към всеки участник се поддържа един или няколко EOSIO акаунта с техните асоциирани публични ключове за достъп до EOSIO, както и публичните ключове, които те са предоставили за размяна на криптирани съобщения.

Process Engine следи обработката и данните, свързани с конкретни многостъпкови SDM процеси. Този модул може да създава нови SDM процеси и да променя статусите на процесите и потребителските роли по време на изпълнение на процесите зависещи от по-сложни, дефинирани от самите участници правила.

Той също така отговаря за автоматичните плащания в резултат на ръчно взаимодействие или задействани автоматично при достигане на зададен краен срок. При наличие на сложни процеси, които могат да се състоят от няколко гъвкаво избрани

стъпки и да включват участие на конкурентни участници, изниква нуждата от гъвкав оркестратор като модул за управление на процесите.

Един примерен сценарий е, когато клиентът търси подходящи консултанти, които да извършат технически LM процеси по прозрачен и проверим начин. Използвайки process engine могат да се зададат параметрите на процеса, който клиентът има нужда да се извърши по дадена операция – инсталация, ъпгрейд или други. Когато параметрите са заложени, различни консултантски фирми могат да правят оферти за извършване на дейността. Различни параметри на офертите мога да бъдат време и цена за извършване на операцията, време на неналичност на ключовите системи, гаранции и KPI за извършване на дейността и т.н.

Друга ситуация, при която има необходимост от гъвкав Process Engine, е когато зададената операция се състои от няколко различни подоперации, които трябва да се извършат едновременно или последователно. При такава задача ролята на Process Engine е да проследи изпълнението на всяка от стъпките и да уведомява участниците, които имат отговорност за изпълнението на всяка от следващите стъпки.

Когато комуникацията или уведомленията между участващите страни трябва да бъдат криптирани, те използват функционалността за защитена комуникация (**Secure Notification**). Основна задача на този интелигентен договор е да осигурява достъп до криптираните данни на тези участници, които трябва да са способни да ги разчетат. Когато се криптира информация, съдържаща малък обем данни най-подходящо е да се използва публичния ключ на получателя. В случай на криптиране на информация, съдържаща голям обем данни, начинът да се защити е чрез генериран за нея симетричен ключ. След това самият симетричен ключ трябва да се подпише от участника, който предоставя информацията и да се криптира с публичните ключове на участниците, които имат право да декриптират тази информация. По този начин всеки участник, който има право да чете тази информация може да използва собствения си транспортен частен ключ за да прочете ключа за криптираната информацията, както и чрез публичния ключ на предоставилия информацията да се увери, че данните са автентични.

Интелигентният договор на Software Marketplace съхранява информация за налични SDM артефакти и версии предлагани от разработчиците на софтуер. Състоянието на Software Marketplace се актуализира от *oracles*, работещи при доставчика на софтуер и се използва за автоматично валидиране на артефакти,

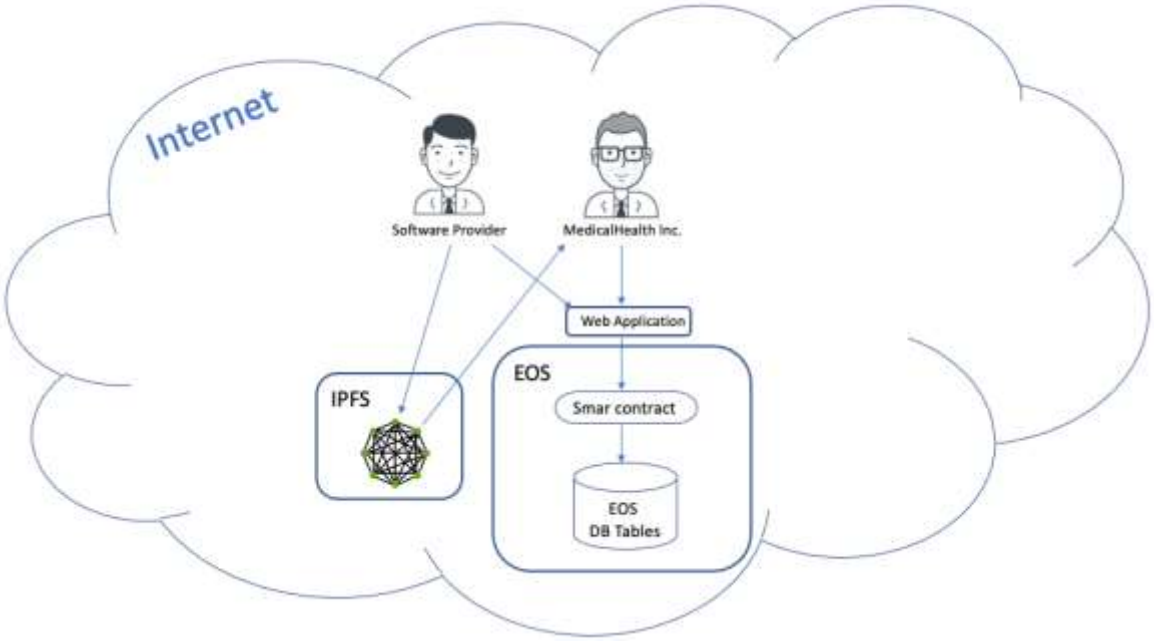
инсталирани при клиента и проверки за корекции и препоръки за надграждане. Информацията, която се съхранява в този модул е разпределена между EOSIO и IPFS. Общодостъпните данни могат да бъдат съхранявани в некриптиран вид - информация за наличните продукти предлагани от доставчиците на софтуера, техните версии и статуса на техните версии (стабилни, неподдържани, бета-разпространение), и т.н. Всеки продукт или компонент на продукт най-често се съхранявани в IPFS поради размера си. В някои специални случаи е възможно в IPFS да се съхраняват данни за версии на софтуера или техни корекции, които не са общодостъпни за всички, например корекция по сигурността. Една част от софтуерния пазар е достъпна само за доставчика на софтуера. Тази част съдържа информация за клиентите, използващи даден софтуер – версии и използвани компоненти на конкретни системи за всеки един техен клиент. По този начин при започване на всеки SLM процес данните от софтуерния пазар, като начална и крайна версия на продукта при ъпгрейд или кои нови компоненти трябва да се инсталират и конфигурират за работа, автоматично могат да се използват за стартиране на нов SDM процес от клиентите на софтуера. Данните в индивидуалните секции на всеки софтуерен доставчик могат да бъдат интегрирани в техните CI/CD процеси, за да могат бързо и гъвкаво да предоставят актуални версии на клиентите си. Допълнителни задачи, които могат да се изпълняват от този модул са проследяване на периодите, за които софтуера е лицензиран, както и нотификации при изтичане на периода на лицензиране или изтичане на периода на поддръжка на версията на софтуера, който се използва от конкретния клиент.

И накрая, всички данни от клиентските системи се събират от интелигентния договор *Track LM/KPI Data* или чрез ръчни операции или чрез установени *oracles*, работещи от страна на клиента. Самите потенциално поверителни данни в крайна сметка се криптират и качват в IPFS. Ключовете за дешифриране на данни във всеки файл се генерират и записват в информацията в Track LM/KPI Data за всяка страна. Данните, съхранявани в този модул, се събират изключително при работата на клиентските системи. Събираната информация е изключително ценна за доставчиците на софтуера, тъй като може да се използва за получаване на обратна връзка при всеки от SDM процесите, които се извършват през предложената система. Данни за времето на изпълнение и евентуални трудности и забавяния при реални изпълнения на SDM процеси позволяват да се дават по-прецизни предвиждания за изпълнението на тези процеси при бъдещите клиенти, както и да се предупреждава за рискове при

изпълнението им. Друго предимство на събирането на данни за изпълнение на нормалните процеси е обратната връзка към доставчика на софтуера за качеството на предлагания от тях софтуер. Метрики като натоварване на системите, наличното количество процесорна мощ, памет и мрежови ресурси мога да се използват, за да се анализира начина, по който клиентите на софтуера го използват реално. Събраните данни могат да се анализират, както с предварително зададени алгоритми, така и с интегрирани модули за изкуствен интелект, в които да правят по-точни оценки.

Глава 4. Прототип

Тази глава съдържа описание на имплементацията на базов прототип, покриващ примерен сценарий за медицински софтуер, който използва архитектурата, дефинирана в предходната глава. Участниците в сценария общуват чрез децентрализирано EOSIO приложение и IPFS.



В сценария компанията доставчик (Software Provider) доставя корекция (patch) на софтуера, свързана със сигурността на системата. Тази корекция първоначално се доставя по сигурен начин само до клиента MedicalHealth Inc., който в момента използва проблемната версия на продукта. След нейното валидиране новата версия става достъпна за всички клиенти.

Технологии използвани за целите на прототипа и неговото реализиране:

- EOSIO Quickstart Web IDE – избора на среда за разработка е изключително важен етап при създаването на софтуер. Следващата глава съдържа описание на средата, както и какви са предимствата на разработване на EOSIO компоненти в Web IDE.
- EOSIO Contract Development Toolkit или EOSIO.CDT – библиотека за разработване на умни договори. В EOSIO умните договори се разработват на C++. Без значение кой е езикът за програмиране това не е възможно без необходимите библиотеки, които да улеснят разработката и да елиминират дублирането на код при програмистите в различни проекти.
- EOSJS – има същите цели както EOSIO.CDT, но с фокус върху разработката на потребителски интерфейс. EOSJS дава всичко необходимо, за да може чрез JavaScript да се извикват функции в умните договори, достъпни в блокчейна.
- EOS-Encrypt –библиотека даваща възможност на ниво потребителски интерфейс и JavaScript код да се криптира и декриптира информацията в блокчейна.
- IPFS и IPFS Desktop – IPFS е много гъвкава, сигурна и подходяща файлова система за целите на прототипа. Ще се опише как най-популярното приложение IPFS Desktop помага в реализацията на сценария демонстриращ прототипа.

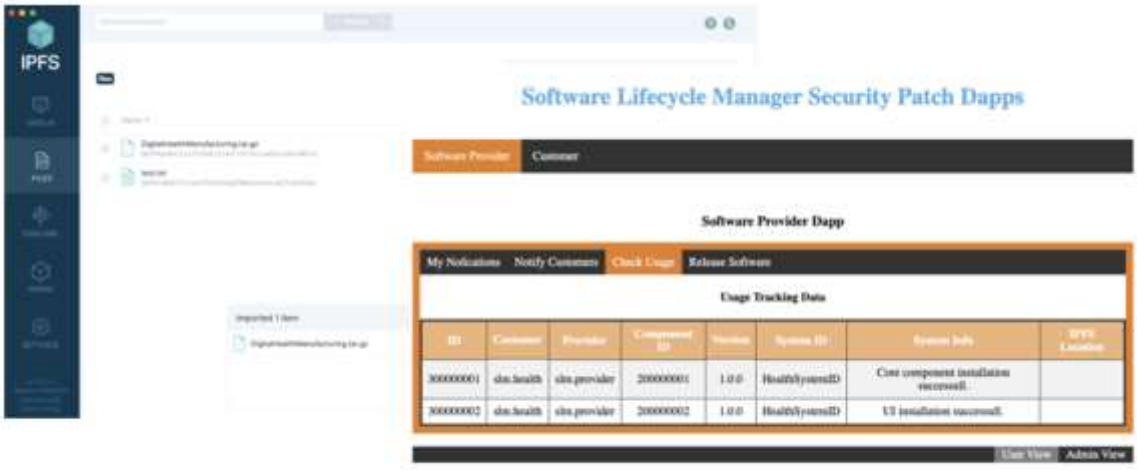
Като част от практическата реализация на прототипа са разгледани и разработените следните основни компоненти:

- Четири умни договора – Те реализират бизнес функционалността на системата като съхраняват данните в мултииндекс таблици в блокчейн мрежата.
- Инсталационна процедура – Предоставя необходимите инструменти за инсталиране на умните договор в блокчейна.
- Уеб базиран потребителски интерфейс – Имплементацията следва установените практики в света на EOSIO за разработка на уеб интерфейси, като ползва ReactJS, EOSJS и EOS-Encrypt.

В прототипа е разгледан сценарий, в който софтуерна компания открива проблем със сигурността в неин продукт. Стъпките, които са проследени след настъпване на това събитие са: качване на подобрената версия на продукта в IPFS, криптирано известяване

на засегнатите клиенти, прилагане на поправката при клиентите и в края публично известяване до всички, както и официално публикуване на новата версия на продукта.

За осъществяване на стъпките от сценария са използвани всички умни договори, както и съхраняване на информация в IPFS. Всички възможни комуникации между модулите, както и криптиране на съобщенията са тествани. Потребителските екрани демонстрират последователността на сценария и коректното известяване и съхраняване на информация.



Кода на прототипът може да бъде намерен тук https://github.com/bisertzvetkov/my-eosio-web-ide/tree/master

Прототипът демонстрира успешното изпълнение на всички цели, заложени в предложената иновативна SDM блокчейн базирана архитектура.

Глава 5. Заключение

Обобщение

Живеем в десетилетие белязано от думите блокчейн и биткойн. Десетилетие, в което криптовалутата разтърси финансовите пазари и промени възможностите за инвестиция. Десетилетие, което дава обещание, че криптографията и блокчейн платформите са на път да премахнат централните монополни институции и да създадат свят на демокрация

в p2p мрежи, на справедлив и надежден процес за съхраняване и прехвърляне на блага между различни участници.

Именно затова дисертационният труд е насочен към тази актуална технология. Технология, обещаваща да революционизира всеки от познатите ни днес бизнеси и процеси. Технология, която дава заявка за намаляване или отпадане на нуждата от намеса на скъпо платени специалисти от професии като нотариус или банкер. В която много участници ще могат да се включат в споделени процеси, като ще се запази отговорността, собствеността и коректността на всеки един участник.

В последните години са изследвани различни класически бизнеси от финанси, през имоти до автономни коли и застраховки като темата покрита от дисертационния труд, а именно SDLC е все още неизследвана. Ето защо приноса на този труд е уникален и допринася за налагането на блокчейн като нов стандарт в качествените и надеждни SDM услуги.

Още повече представената блокчейн SDLC архитектура, която решава проблемите с видимост на данните, прогнозиране на рисковете, дефинирането на ясни отговорности в SDM, би могла да се разшири и да покрие други области от заобикалящия ни свят.

Прототипът, който много детайлно е имплементиран и описан може да служи за база и на други сходни решения. Умните договори и комуникационните протоколи биха могли да се променят спрямо нуждите на всяка една индустрия.

Възможното развитие на обследваната област и на дефинираната блокчейн архитектура съдържа огромен потенциал за видоизменяне в посока решаване на сходни проблеми.

Научни приноси на дисертацията

- Направен е детайлен анализ на съществуващите решения за управление на жизнения цикъл на бизнес софтуерни системи и какви са трудностите, уязвимостите и нерешените проблеми в тази област.

- Формулирани са направления и функционалности, които са от важно значение за дизайн на предложена нова SDLC блокчейн система.

- Описани са характеристики на различните съществуващи платформи и анализ на това как техните специфики биха могли да адресират предизвикателствата в SDLC областта. За избраната платформа е направен детайлен анализ на спецификите и функционирането на тази платформа.

- Създаден е иновативен SDLC блокчейн дизайн и е показано как той преодолява съществуващите класически SDLC предизвикателства.

- Представен е прототип, базиран на иновативната архитектура, като детайлно е описана средата в която той работи: интерфейси, умни договори, модули и агенти, комуникационни протоколи.

- Проследени са ключови сценарии с поетапното им изпълнение адресиращо SDLC предизвикателствата.

- Обобщени са резултатите от разработката на новата SDLC система и са направени анализи на различните сценарии, успешно адресирани от иновативната архитектура. Демонстрирани са предимствата особено в областта на проследяемост, сигурност и дефиниране на отговорности.

Апробация на резултатите

Резултатите от дисертацията са публикувани в следните статии:

1. Tsvetkov B., Kostadinov H., DLT Smart Contract Platforms for Software Lifecycle Management, AIP Conference Proceedings,Vol. 2164, art. n. 120015, 2019, Scopus, SJR: 0.190
2. Tsvetkov B., Kostadinov H., Modern Software Lifecycle Management Leveraging the Power of Blockchain, Proceedings of International Workshop on Algebraic and Combinatorial Coding Theory, pp. 145- 149, 2020, Scopus
3. Tsvetkov B., Kostadinov H., Using DLT in Software Lifecycle Management, Studies in Computational Intelligence,Vol. 961, pp. 393 – 404, 2021, Springer, Scopus, SJR: 0.215
4. Tsvetkov B., Kostadinov H., Software Lifecycle Based on DLT, Communications in Computer and Information Science, Springer, Scopus, SJR: 0.160, to appear

Доклади на научни форуми

Получените резултати са представени лично на следните научни форуми:

1. Tsvetkov B., Kostadinov H., Blockchain technology, Национален семинар по Теория на кодирането “Професор Стефан Додунеков“, 30.11 - 03.12.2017г.
2. Tsvetkov B., Kostadinov H., How Blockchain is solving typical Software Lifecycle Management problems/issues, Национален семинар по Теория на кодирането “Професор Стефан Додунеков“, 8 - 11.11.2018г.
3. Tsvetkov B., Kostadinov H., Using DLT in Software Lifecycle Management, Annual Meeting of the Bulgarian Section of SIAM, 18 - 20.12.2018г.
4. Tsvetkov B., Kostadinov H., DLT Smart Contract Platforms for Software Lifecycle Management, 11th International Conference for Promoting the Application of Mathematics in Technical and Natural Sciences, AMiTaNS 2019, 20 - 25.06.2019г.
5. Tsvetkov B., Kostadinov H., Modern Software Lifecycle Management Levareging the Power of Blockchain, International Workshop on Algebraic and Combinatorial Coding Theory, 11 -17.10.2020г.
6. Tsvetkov B., Kostadinov H., Software Lifecycle Based on DLT, Third Conference on Digital Transformation, Cyber Security and Resilience, 29.09 - 01.10.2021г.

Благодарности

Историята е показала, че при внедряване на иновация винаги водеща роля са имали научните институции, чиято функция е да обследват, анализират и съдействат за изясняване приложимостта на новата тема в реалния живот. Както и че всяко едно изследване, независимо от обема и обхвата му, е крачка напред в посока на приложимост с цел улеснение на хората и бизнесите за ползване на по-качествени и ефективни процеси и услуги.

Именно затова бих искал да благодаря на ИМИ към БАН за предоставената ми възможност да съм част от техния талантлив колектив и да дам и аз своя скромен принос към налагането на блокчейн като нов висококачествен стандарт в революцизирането на класическите бизнес процеси.

Благодарности и към моите колеги Живко Желязков и Георги Константинов за ползотворното време прекарано заедно в дискусии и генериране на идеи. В очакване те да продължат работата по темата и в близко бъдеще да представят надградени разработки.

Не на последно място, бих искал да благодаря на моя научен ръководите Христо Костадинов за неуморната му подкрепа по време на моето следване, за безценните му насоки относно етапите за реализиране на научна дейност, за помощта при дефиниране на цели и демонстриране на резултати, както и за начина на публикуване на доклади и статии.

Библиография

1. Imran Bashir (2018), “Mastering blockchain: distributed ledgers, decentralization and smart contracts explained”, Packt Publishing; 2nd Revised edition

2. S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system,” 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf

3. Paul A. Tatro (2018) “Blockchain Unchained: The Illustrated Guide to Understanding Blockchain” , Book Counselor LLC

4. Elsden, C., Manohar, A., Briggs, J., Harding, M., Speed, C., Vines, J. “Making sense of blockchain applications: a typology for HCI.” In: CHI 2018. ACM (2018)

5. Z. Zheng, S. Xie, H. N. Dai, X. Chen, and H. Wang, ‘‘Blockchain challenges and opportunities: A survey,’’ Int. J. Web Grid Services, vol. 14,no. 4, pp. 352–375, 2018

6. Basit Shahzad , Jon Crowcroft , “Trustworthy Electronic Voting Using Adjusted Blockchain Technology.”, IEEE Access 7: 24477-24488

7. Bhardwaj, S., Kaushik, M. “Blockchain—technology to drive the future.” In: Satapathy, S.C., Bhateja, V., Das, S. (eds.) Smart Computing and Informatics. SIST, vol. 78, pp. 263–271. Springer, Singapore (2018).

8. Suhaliana bt Abd Halim, N., Rahman, M.A., Azad, S., Kabir, M.N.” Blockchain security hole: issues and solutions.” In: Saeed, F., Gazem, N., Patnaik, S., Saed Balaid, A.S., Mohammed, F. (eds.) IRICT 2017. LNDECT, vol. 5, pp. 739–746. Springer, Cham (2018).

9. Z. Zheng, S. Xie, H. Dai, X. Chen and H. Wang, "An overview of blockchain technology: Architecture consensus and future trends", Proc. IEEE Int. Congr. Big Data (BigData Congr.), pp. 557-564, Jun. 2017.

10. Daniel Drescher (2017) “Blockchain Basics A Non-Technical Introduction in 25 Steps”, ISBN: 978-1-4842-2604-9

11. UNITED STATES NUCLEAR REGULATORY COMMISSION (2002), “System Development and LifeCycle Management (SDLCM) Methodology”, Handbook, Version 2.3

12. R. Turpin , “A progressive software development lifecycle”, Proceedings of ICECCS '96: 2nd IEEE International Conference on Engineering of Complex Computer Systems

13 Gagan Gurung, Rahul Shah, Dhiraj Prasad Jaiswal, “Software Development Life Cycle Models-A Comparative Study”, International Journal of Scientific Research in Computer Science Engineering and Information Technology

14. Shylesh S. (2017), “A Study of Software Development Life Cycle Process Models.” SSRN Electronic Journal

15. Andreas M. Antonopoulos “Mastering Bitcoin”, 2nd Edition, Publisher(s): O'Reilly Media, Inc.

16. Sudhan, A., & Nene, M. J. (2018). “Peer Selection Techniques for Enhanced Transaction Propagation in Bitcoin Peer-to-Peer Network”, 2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS)

17. Matthew Underhill , “The Bitcoin Book: A Beginner's Guide to the Future of Finance”, Independently published (September 21, 2020)

18. Zhu, F., Chen, W., Wang, Y., Lin, P., Li, T., Cao, X., & Yuan, L. (2017). “Trust your wallet: A new online wallet architecture for Bitcoin”, 2017 International Conference on Progress in Informatics and Computing (PIC).

19. D. Larimer, EOS.IO Technical White Paper

20. EOSIO resource, https://eos.io

21. Zheng, W., Zheng, Z., Dai, H.-N., Chen, X., & Zheng, P. (2021). “XBlock-EOS: Extracting and exploring blockchain data from EOSIO”, Information Processing & Management, 58(3), 102477.

22. IPFS resource, https://ipsf.io

23. Jianjun, S., Ming, L., & Jingang, M. (2020), “Research and application of data sharing platform integrating Ethereum and IPFs Technology”, 2020 19th International Symposium on Distributed Computing and Applications for Business Engineering and Science (DCABES).

24. Guidi, B., Michienzi, A., & Ricci, L. (2021), “Data Persistence in Decentralized Social Applications: The IPFS approach” , 2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC).

25. Sudhani V. , Divakar Y. , Girish C.,“ Introduction of Formal Methods in Blockchain Consensus Mechanism and its Associated Protocols“, IEEE Access

26. Parma Bains , “Blockchain Consensus Mechanisms: A Primer for Supervisors” , January 2022

27. Lashkari, B., & Musilek, P. (2021), “A Comprehensive Review of Blockchain Consensus Mechanisms”, IEEE Access, 9, 43620–43652.

28. Yusoff, J. , Mohamad, Z. and Anuar, M. (2022), “A Review: Consensus Algorithms on Blockchain”, Journal of Computer and Communications, 10, 37-50

29. Casino, F., Dasaklis, T.K., Patsakis, C. “A systematic literature review of blockchain-based applications: Current status, classification and open issues.” Telematics and Informatics 2019

30. lnes, S., Jansen, A., “Blockchain technology as s support infrastructure in e-government”, In: Janssen, M., Axelsson, K., Glassey, O., Klievink, B., Krimmer, R., Lindgren, I., Parycek, P., Scholl, Hans J., Trutnev, D. (eds.) EGOV 2017. LNCS, vol. 10428, pp. 215–227. Springer, Cham (2017)

31. García-Bañuelos, L., Ponomarev, A., Dumas, M., Weber, I., “Optimized execution of business processes on blockchain” In: Carmona, J., Engels, G., Kumar, A. (eds.) BPM 2017. LNCS, vol. 10445, pp. 130–146. Springer, Cham (2017).

32. Bocek, T., Rodrigues, B., Strasser, T., Stiller, B., “Blockchains everywhere - a use-case of blockchains in the pharma supply-chain” In: 2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM) (2017)

33. Shae, Z., Tsai, J., “On the design of a blockchain platform for clinical trial and precision medicine” In: 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS) (2017)

34. Toyoda, K., Mathiopoulos, P., Sasase, I., Ohtsuki, T., “A novel blockchain-based Product Ownership Management System (POMS) for anti-counterfeits in the post supply chain”, IEEE Access 5, 17465–17477 (2017)

35. Munsing, E., Mather, J., Moura, S., “Blockchains for decentralized optimization of energy resources in microgrid networks”, In: 2017 IEEE Conference on Control Technology and Applications (CCTA) (2017)

36. Kshetri, N.,” Blockchain’s roles in strengthening cybersecurity and protecting privacy. Telecommun”, Policy 41, 1027–1038 (2017)

37. Aitzhan, N.Z., Svetinovic, D., “Security and privacy in decentralized energy trading through multi-signatures, blockchain and anonymous messaging streams”, IEEE Trans. Dependable Secure Comput. 1 (2016)

38. Grumbach, S., Riemann, R., “Distributed random process for a large-scale peer-to-peer lottery”, In: Chen, L.Y., Reiser, H.P. (eds.) DAIS 2017. LNCS, vol. 10320, pp. 34–48. Springer, Cham (2017)

39. Wijaya, D.A., Liu, J.K., Suwarsono, D.A., Zhang, P., “A new blockchain-based value-added tax system”, In: Okamoto, T., Yu, Y., Au, M.H., Li, Y. (eds.) ProvSec 2017. LNCS, vol. 10592, pp. 471–486. Springer, Cham (2017)

40. Buterin V.: A Next-Generation Smart Contract and Decentralized Application Platform (2013). https://github.com/ethereum/wiki/wiki/White-Paper

41. Canessane, R. A., Srinivasan, N., Beuria, A., Singh, A., & Kumar, B. M. (2019), “Decentralised Applications Using Ethereum Blockchain”, 2019 Fifth International Conference on Science Technology Engineering and Mathematics (ICONSTEM)

42. C. Dannen, “Introducing Ethereum and Solidity”, Berkeley, CA:Apress, 2017.

43. Zishan Zhao, “Comparison of Hyperledger Fabric and Ethereum Blockchain”, 2022 IEEE Asia-Pacific Conference on Image Processing, Electronics and Computers (IPEC)

44. IBM Hyperledger Fabric. Retrieved from https://www.ibm.com/blockchain/hyperledger

45. Androulaki, E., Manevich, Y., Muralidharan, S., Murthy, C., Nguyen, B., Sethi, M., Laventman, G. (2018), “Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains”, Proceedings of the Thirteenth EuroSys Conference on - EuroSys ’18.

46. Debajani Mohanty, “R3 Corda for Architects and Developers: With Case Studies in Finance, Insurance, Healthcare, Travel, Telecom, and Agriculture”, Apress; 1st ed. edition (June 29, 2019)

47. Khan, C., Lewis, A., Rutland, E., Wan, C., Rutter, K., & Thompson, C. (2017), “A Distributed-Ledger Consortium Model for Collaborative Innovation”, Computer, 29–37.

48. M. Hearn, “Corda: A Distributed Ledger white paper R3”, Nov. 2016, Available: docs.corda.net/_static/corda-technical-whitepaper.pdf

49. Hewa, T. M., Hu, Y., Liyanage, M., Kanhare, S. S., & Ylianttila, M. (2021), “Survey on Blockchain-Based Smart Contracts: Technical Aspects and Future Research”, IEEE Access, 9, 87643–87662.

50. Khan, S. N., Loukil, F., Ghedira-Guegan, C., Benkhelifa, E., & Bani-Hani, A. (2021), “Blockchain smart contracts: Applications, challenges, and future trends”, Peer-to-Peer Networking and Applications, 14(5), 2901–2925

51. Al-Breiki, H., Rehman, M. H. U., Salah, K., & Svetinovic, D. (2020), “Trustworthy Blockchain Oracles: Review, Comparison, and Open Research Challenges”, IEEE Access, 8, 85675–85685