ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

ІМЕНІ ІВАНА ПУЛЮЯ

**ТХІР ОКСАНА СЕРГІЇВНА**

УДК 004.75

**ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ BLOCKCHAIN В ЗАДАЧАХ СТРАХУВАННЯ**

122 « Комп’ютерні науки »

# Автореферат

дипломної роботи на здобуття освітнього ступеня «магістр»

Тернопіль – 2018

Роботу виконано на кафедрі комп’ютерних наук Тернопільського національного технічного університету імені Івана Пулюя Міністерства освіти і науки України

|  |  |
| --- | --- |
| **Керівник роботи:** | кандидат технічних наук, доцент  **Фриз Михайло Євгенович,**  Тернопільський національний технічний університет імені Івана Пулюя, доцент кафедри комп’ютерних наук |
| **Рецензент:** | кандидат технічних наук, доцент  **Гладьо Юрій Богданович,**  Тернопільський національний технічний університет імені Івана Пулюя, доцент кафедри інформатики і математичного моделювання |

Захист відбудеться 27 грудня 2018 р. о 900 годині на засіданні екзаменаційної комісії № 30 у Тернопільському національному технічному університеті імені Івана Пулюя за адресою: 46001, м. Тернопіль, вул. Руська 56, навчальний корпус №1, ауд.702

Секретар екзаменаційної комісії №30 Литвиненко Я.В.

**ЗАГАЛЬНА ХАРАКТЕРИСТИКА РОБОТИ**

**Актуальність теми роботи** зумовлена відсутністю вільнорозповсюджуваних рішень для вирішення задачі дотримання умов страхового договору, оскільки іноді відбуваються певні спекуляції при настанні страхового випадку.

Розроблюваний smart-контракт буде забезпечувати чесні відшкодування, шляхом існування сертифікованого сервісного центру який не буде мати жодних даних про власника машини, буде здійснювати оцінку рівня відшкодувань та при потребі проводити ремонт ТЗ. Якщо, клієнт вважатиме нарахований рівень відшкодувань низьким, може не погодитися та знову пройти процедуру оцінки.

**Мета роботи:** дослідитиможливості технології Blockchain в страховій галузі та забезпечити дотримання умов страхового договору.

**Задачі досліджень:** створення системи для забезпечення непідробності даних та виконання умов договорів в страховій галузі.

**Об’єкт досліджень:** процес впровадження технології Blockchain для задач страхування.

**Предмет досліджень:** алгоритм роботи smart-контракту.

**Методи досліджень:** розробка smart-контракту з використанням Remix IDE та розміщення його на платформі Ethereum.

**Наукова новизна отриманих результатів:** отримано інноваційну методику при роботі з страховими договорами та забезпечується цілісность даних і анонімность клієнта при оцінці розміру відшкодування, надання швидкої грошової компенсації після настання страхового випадку

**Практичне значення отриманих результатів.**

Отримані в роботі результати доцільно використати для страхових компаній, які хочуть покращити процес страхових виплат, знизити рівень шахрайства як з сторони працівників, так і з сторони клієнтів, зменшити масштаби документообігу, знизити кількість найманих працівників, збільшити довіру клієнтів за рахунок забезпечення прозорості роботи та непідробності договорів.

**Апробація.** Результати досліджень доповідались на VІI Міжнародній науково-технічній конференції молодих учених та студентів «Актуальні задачі сучасних технологій», Тернопіль, ТНТУ, 28-29 листопада 2018 р. Матеріали роботи опубліковано в збірнику тез цієї конференції.

**Структура роботи.** Робота складається з розрахунково-пояснювальної записки та мультимедійної презентації. Розрахунково-пояснювальна записка містить вступ, 8 розділів, висновки та список використаної літератури. Загальний обсяг текстової частини – 138 аркушів формату А4, яка включає 6 таблиць, 53 рисунків. Мультимедійна презентація містить 26 слайдів.

# ОСНОВНИЙ ЗМІСТ РОБОТИ

У **вступі** подано загальну характеристику роботи, сформульовано основні завдання, які необхідно вирішити для розкриття теми роботи, актуальність роботи і практичну значимість отриманих результатів.

У **першому розділі «Аналітичний огляд літературних джерел та інших джерел»** проведено аналіз українських та іноземних літературних джерел, обґрунтовано актуальність роботи. Знайдено потенційні застосування технології в фінансовій, медичні, страхові, автомобільні, мобільних додатках, IoT та інших галузях. На основі проведеного літературного огляду та аналізу теоретичних джерел можливостей технології Blockchain зроблено висновок, що для вирішення задачі дотримання умов страхового договору покищо немає вільнорозповсюджуваних рішень. Тому після вирішення проблеми фракталів із мережею Ethereum та незначної адаптації написаний у цій роботі smart-контракт може використовувати будь яка страхова компанія для укладання договорів із клієнтами. Нажаль в сучасних умовах невеликої популярності використання технології блокчейн в сучасній банківській системі, виникає так звана проблема фракталів, коли технологія не може бути інтегрована із класичною моделлю фінансових розрахунків. Проте, в майбутньому технологія може допомогти вирішити багато завдань, але для цього потрібно, щоб вона стала більш поширеною та були прийняті певні законодавчі акти.

У **другому розділі «Системний аналіз технології Blockchain»** здійснено опис технології Blockchain, представлено еволюцію технології, різновиди систем керування (централізовані, децентралізовані, розподілені), типи користувачів (майнери, прості користувачі), приватні та публічні ключі. Здійснено детальний опис генерації приватного ключа із випадкового числа, криптографію на еліптичних кривих. Описано блок транзакцій та забезпечення цілісності даних, криптографічні хеш-функції, проблеми безпеки технології Blockchain, представлено SWOT-аналіз технології Blockchain для страхової галузі.

**У** **третьому розділі «Методи впровадження Blockchain в страхуванні»** здійснено огляд можливих застосувань технології в страхуванні (зменшення експлуатаційних витрат та покращення рівня обслуговування клієнтів, введення даних/перевірка особистих даних, преміальне обчислення/оцінка ризиків/попередження шахрайства, плата за користування, мікрострахування, однорангове страхування). Представлено завдання для яких можна використати технологію в страховій галузі, здійснено огляд платформ для розміщення smart-контрактів (Ethereum, EOS, Waves, Cardano, Stellar, NEO, Hyperledger Fabric, Aeternity, Emotiq, Bitcoin, Side Chains, NXT). Показано класифікацію smart-контрактів (за середовищем виконання, процесу задання і виконання умов, за способом ініціювання, за рівнем приватності), описано аккаунти в Ethereum (аккаунт користувача (гаманець) та аккаунт контракту), показано структуру smart-контракту в Solidity.

**У** **четвертому розділі «Практична реалізація»** розглянуто весь цикл розробки смарт-контракту: вибору необхідних даних, формування вимог, написання програмного коду контракту, компіляції та тестування контракту в інтегрованому середовищі розробки Remix, створення Ethereum гаманця та розгортання контракту в тестнеті блокчейну. Після розгортання транзакцію можна відслідковувати за допомогою оглядача блоків (рис. 1).

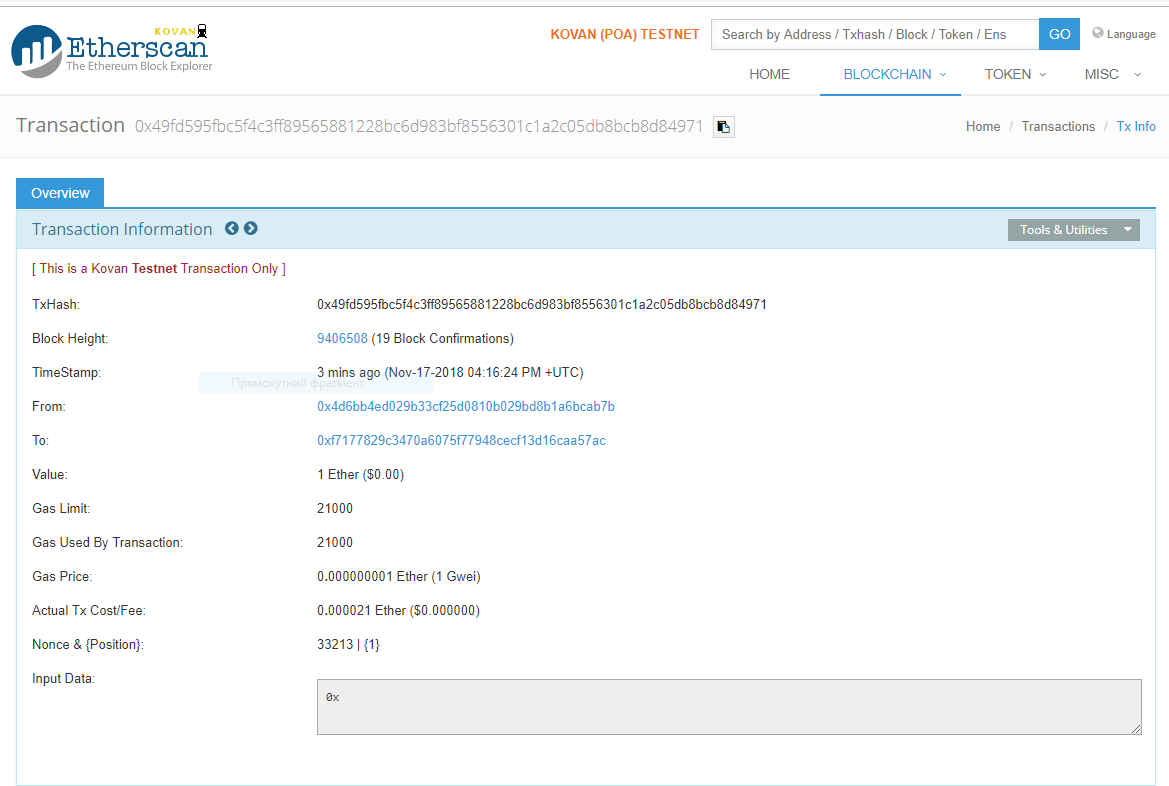


Рисунок 1 – Відслідковування транзакції за допомогою Etherscan

Контракт на Solidity – це набір коду (його функції) і даних (його стан), який знаходиться на конкретній адресі в блокчейн Ethereum. При написанні програмного коду в першому рядку слід вказати версію компілятора – це має гарантувати, що він не буде вести себе по-іншому з новою версією компілятора. Ключове слово pragma, називається так, тому що pragma – це інструкція для компілятора про те, як обробляти код. Коли контракт створений, його конструктор (функція, назва якої збігається з назвою контракту) викликається тільки один раз. Конструктор не обов'язковий. Дозволяється використовувати тільки один конструктор, і це означає, що перевантаження в Solidity не підтримується. Внутрішні аргументи конструктору передаються в ABI закодованими після коду самого контракту.

Якщо smart-контракт повинен створювати інший контракт, то вихідний код (і бінарний) створюваного повинен бути відомий батьківському контракту.

Для оголошення глобальних змінних було вибрано такі значення:

* address public agencyAddr – адреса страхового агента;
* address public workshopAddr – адреса сертифікованої майстерні для оцінки завданих збитків;
* uint public lastPackage – ідентифікатор останнього страхового пакету (послуги);
* uint public lastPolicy – ідентифікатор останнього укладеного страхового полісу;
* uint public lastEvent – ідентифікатор останнього страхового випадку;
* uint public constant timeToPay = 14 days – часова мітка, виділена на оплату купленого полісу (є константою і не може бути змінена в процесі роботи контракту);
* enum AllowedPeople – список, який складається з значень one, many і визначає на яку кільість осіб поширюється дія полісу;
* enum PolicyStatus – список, який вказує на статус полісу – waitClient (очікує на оплату від клієнта), activePolicy (активний поліс), endedPolicy (поліс із закінченим терміном дії), unpaid (неоплачений поліс, термін на оплату якого вийшов);
* enum EventStatuses – список, який визначає статус страхового випадку – newEvent (ще не розглянута подія), rejectedEvent (відхилена оцінювачем подія), estimatedEvent (оцінена подія, чекає на рішення клієнта), successEvent (успішна подія), canceledEvent (відхилена клієнтом оцінка).

Змінні, які мають область видимості public є відритим станом контракту можуть бути переглянуті будь-ким в будь-який момент часу. Інші дані є закритим станом і можуть оперуватись лише самим контрактом. Після написання програмного коду smart-контракту потрібно пройти процес компіляції (рис. 2), для того щоб виявити, чи є якісь помилки в програмному коді.

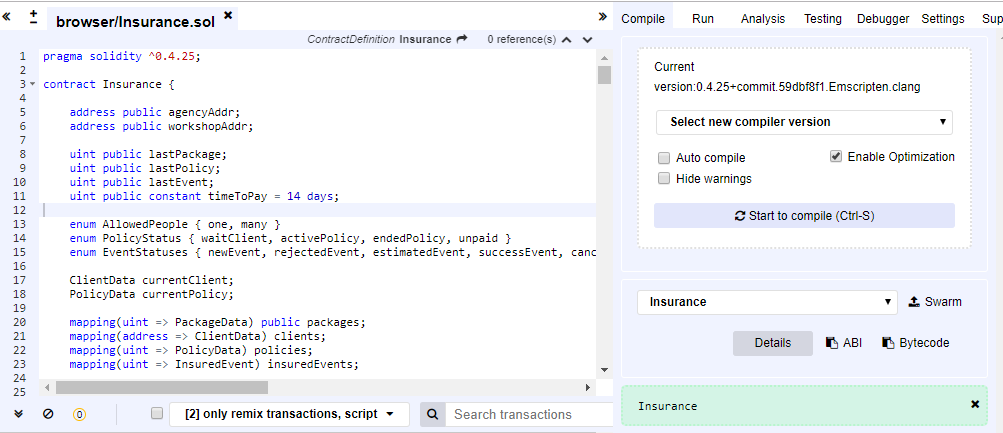


Рисунок 2 – Компіляція smart-контракту Insurance в Remix IDE

Після успішної компіляції відбувається процес розгортання smart-контракту. Для розгортання смарт-контракту вбудованими засобами Remix IDE, на вкладці Run, в полі Environment вибирається Injected Web3 (для взаємодії із підключеним в браузері MetaMask), вказати необхідний gas limit та вартість одиниці gas, обліковий запис з якого буде проводитись розгортання, вибрати скомпільовану версію смарт-контракту і натиснути кнопку Deploy (рис. 3).

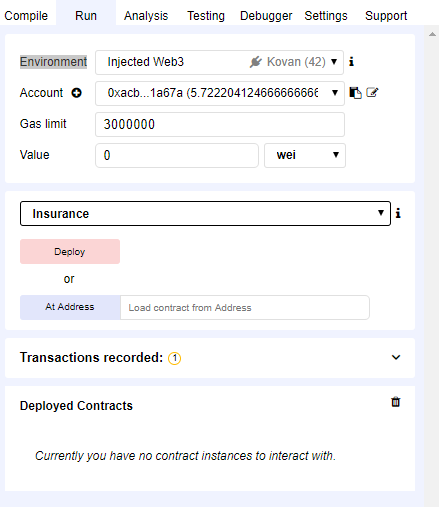


Рисунок 3 – Розгортання smart-контракту засобами Remix IDE

Після цього потрібно перейти в гаманець та підтвердити транзакцію. Після підтвердження транзакції, буде відображено адресу новоствореного контракту. Для наповнення смарт-контракту даними можна проводити за допомогою Remix IDE або за Etherscan (для клієнта). Після кожного додавання інформації потрібно з допомогою MetaMask підтверджувати транзакцію. Після чого виводиться повідомлення про створену транзакцію із посиланням на неї в Etherscan (рис. 4).

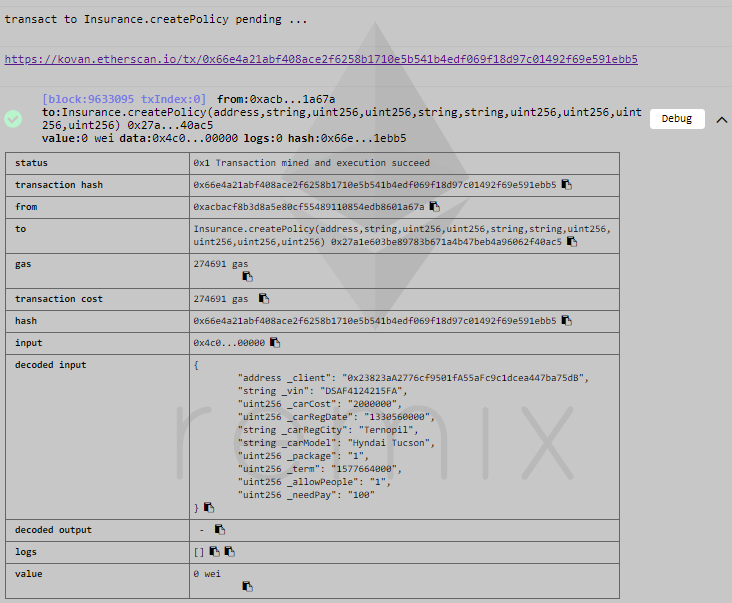


Рисунок 4– Повідомлення в консолі Remix IDE про успішно створену транзакцію

Прочитати вільно інформацію про клієнта може тільки страховий агент або сам клієнт (рис. 5).



Рисунок 5 – Результат виклику геттер-функції із обмеженням доступу

Після настання події, ліцензований сервісний центр надає оцінку вартості роботи. Якщо клієнт погоджується із сумою нарахування, то здійснюється виклик функції і відбувається переказ коштів (рис. 6). В іншому випадку, відбувається повторна оцінка.

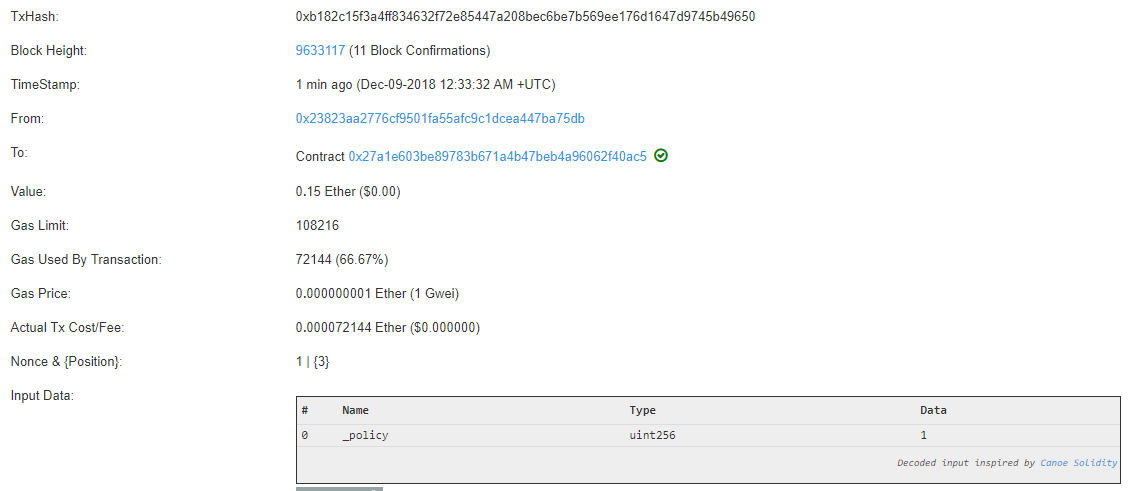


Рисунок 6 – Відображення транзакції виклику функції про оплату страхового полісу

**В п’ятому розділі «Спеціальна частина»** розглянуто фунціонал та можливості інтегрованого середовища розробки. Remix – це потужний, вільно розповсюджуваний інструмент, який призначений для допомоги в написанні smart-контрактів мовою Solidity у вікні браузера. Інтегроване середовище розробки написане за допомогою JavaScript. Remix можна використовувати локально на ПК та онлайн на відповідному сайті за допомогою будь-якого веб-браузера. Це середовище розробки підтримує тестування, налагодження та розгортання smart-контрактів і має багато інших функцій. Завантажити середовище з його всіма функціями можна на сайті remix.ethereum.org. Інструмент IDE доступний в сховищі GitHub. Редактор Remix повторно перекомпільовує код кожного разу, коли змінюється поточний файл або обрано інший файл. Він також забезпечує підсвічування синтаксису. Для зміни розміру шрифту редактора використовуються відповідні піктограми в верхньому лівому куті. Усі відкриті файли Remix відображає як вкладки, і постійно їх зберігає (5 секунд після останніх змін). Усі помилки та попередження компіляції відображаються в боковій колонці, під блоком компіляції.

Наведені нижче параметри дозволяють безпосередньо впливати на виконання транзакції:

* JavaScript VM – всі транзакції будуть виконуватись в тестовому блокчейн-середовищі браузера. Це означає, що нічого не буде збережено, а перезавантаження сторінки перезапускає новий блокчейн з нуля, а попередній не буде збережений;
* Injected Web3 – Remix підключається до ін’єкційного постачальника Web3. Mist та Metamask є прикладом постачальників, які використовують web3;
* Web3 Provider – Remix буде підключатися до віддаленого вузла. Розробнику потрібно вказати URL-адресу вибраного постачальника: geth, parity або будь-якого Ethereum клієнта;
* Account – список облікових записів, пов'язаних із поточним середовищем (та їх балансами);
* Gas limit – максимальний обсяг gas, який можна встановити для всіх транзакцій, створених у Remix;
* Value: вартість для наступної створеної транзакції (після кожного виконання транзакції це значення завжди скидається на 0).

На вкладці «Run» під блоком створення транзакцій знаходиться блок «Initiate Instance», який містить перелік створених договорів та в залежності від наявності контрактів за адресою, наступні операції над ними:

* якщо вказана адреса є екземпляром вибраного контракту, тоді можна взаємодіяти з вже розгорнутою угодою. На цьому етапі немає жодної перевірки, тому потрібно максимально уважно працювати при використанні цієї функції, та переконатися чи є доступ до контракту за цією адресою;
* створити та відправити транзакцію, яка розгортає вибраний контракт. Коли транзакція попаде в видобутий блок, буде додано щойно створений екземпляр (це може зайняти кілька секунд). Якщо конструктор має параметри, їх потрібно вказати.

Перевірка транзакції триватиме кілька секунд. У цей час графічний інтерфейс показує його в режимі очікування. Коли транзакція видобувається, кількість операцій, що очікують на розгляд оновлюється, а операцію додають до журналу.

Metamask – це розширення для браузерів (Chrome, Firefox, Opera, Brave) призначений для швидких платежів, переказів Ethereum і токенів ERC-20 (тобто, відправляти і приймати можна не тільки Ethereum, а ще токени smart-контрактів Ethereum), а також для відправки транзакцій на смарт-контракти та в децентралізовані сервер-клієнтські додатки (DApps). MetaMask не вимагає логіна і не зберігає особисті ключі на сервері, замість цього вони зберігаються в браузері і захищені паролем.

З його допомогою можна відправляти транзакції в мережі Ethereum, встановлюючи Gas limit і Gas price в будь-яких обсягах на свій розсуд. Після відправки транзакції можна переглянути детальну інформацію про неї, зокрема адресу відправника та отримувача, кількість відправленого Ethereum чи токенів, кількість та ціна використаного Gas, дата та час відправки і підтвердження транзакції. Розглянемо переваги та недоліки Metamask, зокрема перевагами є:

* проста схема використання – тобто, зручний інтерфейс допомагає з легкістю як новачкам, так і досвідченим користувачам розуміти послідовність дій;
* захист бази даних – схема захисту продумана по стандартному принципу, є схема відновлення доступу, в разі втрати контролю над сервісом;
* відновлення доступу – проста схема доступу з використання 12 кодових слів;
* можна додатково створювати мульти-акаунти, при цьому немає складних схем повторної реєстрації, достатньо скопіювати в буфер попередню інформацію аккаунта;
* можливість додавати токени стандарту ERC20.

Основними недоліками MetaMask є:

* відсутня двохфакторна аутентифікація;
* обмежений мовний інтерфейс – доступно тільки 4 мовних варіанти – англійська, китайська (спрощений варіант), іспанська, японська.

Etherscan є найпопулярнішим оглядачем блоків (BlockExplorer) для Ethereum Blockchain. BlockExplorer – це, в основному, пошукова система, яка дозволяє користувачам легко здійснювати пошук, підтверджувати та перевіряти транзакції в Ethereum Blockchain. Etherscan не фінансується, не експлуатується, не керується Фондом Ethereum, але натомість існує як незалежна організація. Ethereum Blockchain має публічний реєстр, який показує Etherscan.io, а потім надає цю інформацію через сайт.

Місія Etherscan полягає в представленні прозорості Blockchain шляхом індексації та пошуку всіх транзакцій за Ethereum Blockchain у найбільш прозорий та доступний спосіб. Основними можливостями Etherscan є відслідковування транзакцій, графічна інформація, токени. Можна перевірити статус операції користувача, перевірити адреси та отримати іншу інформацію. В деяких випадках посилання на транзакцію в Etherscan може посприяти прискоренню здійснення угоди. Майнери діють у власних інтересах, тому в першу чергу беруть в обробку транзакції з найбільш високими комісіями. Etherscan не є постачальником послуг електронного гаманця, не зберігає приватні ключі, не контролює транзакції, що відбуваються в мережі Ethereum. Він не надає набір збалансованих надійних служб API, які можуть бути використані для побудови децентралізованих додатків або для подання інформації про Ethereum Blockchain. Децентралізовані біржі зараз знаходяться на стадії бурхливого розвитку. Проблеми з CoinCheck, NiceHash і Binance змушують все більшу кількість користувачів криптовалют переходити на p2p-платформи, де всі учасники наділені рівними правами і не мають можливість управляти чужими коштами

**В шостому розділі «Обгрунтування економічної ефективності»** здійснено обґрунтування економічної ефективності шляхом розрахунку норм часу на виконання науково-дослідної роботи, визначення витрат на охорону праці та відрахувань на соціальні заходи, розрахунку матеріальних витрат та на електроенергію, обрахунку суми амортизаційних відрахувань, обчислення накладних витрат, складання кошторису витрат та визначення собівартості науково-дослідної роботи, розрахунку ціни програмного продукту та визначення економічної ефективності і терміну окупності капітальних вкладень.

**В сьомому розділі «Охорона праці та безпека в надзвичайних ситуаціях»** розглядається електробезпека користувачів ПК та основні принципи та функції управління охороною праці на виробництві.

**В восьмому розділі «Екологія»** екологічному розділі розглянуто абсолютні показники екологічних явищ та енергозбереження і його роль у вирішенні екологічних проблем.

# ЗАГАЛЬНІ ВИСНОВКИ

1. Досліджено предметну область, проаналізовано літературні джерела, описано потенційні застосування технології. На основі аналізу, було здійснено постановку завдання;
2. Побудовано алгоритм роботи smart-контракту, вирішено ролі які має виконувати кожен учасник та їх можливості до доступу до інформації;
3. Здійснено розробку smart-контракту з використанням мови програмування Solidity, розміщення розробленого контракту в блокчейні Ethereum та отримано інноваційну методику роботи із страховими договорами;
4. Для перевірки роботоздатності та відповідності вимогам було проведено тестування розробленого smart-контракту. Під час тестування не було виявлено помилок в роботі, всі передбачені функції виконуються.

# СПИСОК ОПУБЛІКОВАНИХ ПРАЦЬ

1. Тхір О.С. Реалізація smart-контрактів для страхової компанії на платформі Ethereum / O.S. Tkhir Implementation of smart contracts on the Ethereum platform for insurance company // VІI Міжнародна науково-технічна конференція молодих учених та студентів «Актуальні задачі сучасних технологій» ТНТУ імені І. Пулюя – 2018. – с.181-182.
2. Тхір О.С. Інформаційна технологія Blockchain в задачах страхування / O.S. Tkhir Information technology Blockchain in insurance // VІI Міжнародна науково-технічна конференція молодих учених та студентів «Актуальні задачі сучасних технологій» ТНТУ імені І. Пулюя – 2018. – с.183-184.

**ПЕРЕЛІК ОСНОВНИХ ЛІТЕРАТУРНИХ ДЖЕРЕЛ,**

**ВИКОРИСТАНИХ ДЛЯ ДОСЛІДЖЕНЬ**

1. Where Is Current Research on Blockchain Technology?—A Systematic Review Jesse Yli-Huumo, Deokyoon Ko, Sujin Choi , Sooyong Park, Kari Smolander  
   Published: October 3, 2016.
2. Dynamis – if insurance, then Blockchain [Електронний ресурс] // The digital insurer. – 403. – Режим доступу до ресурсу: [https://www.the-digital-insurer.com/blog/insurtech-dynamis-if-insurance-then-Blockchain/](https://www.the-digital-insurer.com/blog/insurtech-dynamis-if-insurance-then-blockchain/) – Дата доступу: 12.06.2018р.
3. Blockchain in Insurance-OpportunityorThreat / J.Lorenz, B. Higginson, P. Olesen, N. Bohlken. – NewYork: McKinsey&CompanyReport, 2016
4. On the Security and Performance of Proof of Work Blockchains [Електронний ресурс]. – 2016. – Режим доступу до ресурсу: <https://www.researchgate.net/publication/309451429_On_the_Security_and_Performance_of_Proof_of_Work_Blockchains/> – Дата доступу: 14.06.2018р.

# АНОТАЦІЯ

**Тхір О.С. Інформаційна технологія Blockchain в задачах страхування** – Тернопільський національний технічний університет імені Івана Пулюя, м.Тернопіль, 2018 р.

Дипломна робота на здобуття освітнього ступеня «магістр» за спеціальністю 122 «Комп’ютерні науки».

В дипломній роботі виконано дослідження та аналіз предметної області та здійснено опис потенційних застосувань технології, та розроблено smart-контракт. За результатами досліджень отримано методику для роботи з страховими договорами де використовуються всі переваги Blockchain.

**Ключові слова:** Blockchain, страхування, Ethereum, smart-контракт, ланцюжок блоків транзакцій, хеш, транзакція, блок.

# ANNOTATION

**Tkhir O.S. Blockchain IT for insurance problems –** Ternopil National Technical University named after Ivan Puluj, Ternopil, 2018.

Diploma thesis on competition of educational degree «master» for the specialty

122 – «Computer Science».

In the thesis the research and analysis of the subject area was made and a description of the potential applications of the technology was made, and a smart contract was developed. According to research results, a methodology for work with insurance contracts is used where all advantages of Blockchain are used.

**Keywords:** Blockchain, insurance, Ethereum, smart-contract, passenger transaction blocks, hash, transaction, block.