МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»**
Кафедра математической кибернетики и компьютерных наук

РАЗРАБОТКА ДЕЦЕНТРАЛИЗОВАННОЙ БИРЖЕВОЙ ПЛАТФОРМЫ

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 411 группы
направления 02.03.02 — Фундаментальная информатика и информационные технологии
факультета КНиИТ
Другаля Артема Олеговича

Научный руководитель
зав. каф. техн. пр., к. ф.-м. н., доцент _____ И. А. Батраева

Заведующий кафедрой
к. ф.-м. н., доцент _____ С. В. Миронов

Саратов 2021

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Каждый день в мире растет число пользователей биржевых платформ. Основное требование к биржевой платформе — возможность доказать, что все происходит в системе честно, и никаких ошибок в процессе торгов не было. К сожалению, бывают случаи, в которых недобросовестные владельцы биржи влияют на процесс торгов. В связи с этим необходимо создавать такие системы, которые бы имели не единственного владельца платформы, а были децентрализованными.

Децентрализованные приложения гибче, прозрачнее, надежнее и имеют более мотивирующую организацию, чем современное программное обеспечение, созданное с применением традиционных моделей.

На сегодняшний день технология блокчейн имеет большой интерес во многих странах мира, например, Эстония, Австралия, Сингапур, Великобритания, США, Япония, Корея, Швейцария, Мальта и в ряде других. Блокчейн может быть полезен, а где-то уже и нашел свое применение не только в криптовалюте, но и в других различных сферах деятельности, таких как: правительство, здравоохранение, финансовые организации, медицина, юриспруденция, логистика, искусство, недвижимость, электроэнергетика, страхование, промышленность, выборы. По оценке Bank of America, глобальный рынок блокчейна может достичь объема в $7 млрд — одномоментно в нем будет задействовано 2% всех серверов в мире [1].

Целью настоящей работы является создание трейдинговой платформы с применением технологии блокчейн.

Были поставлены задачи:

- изучить структуру децентрализованного блокчейн приложения;
- разработать приложение для биржевой платформы с применением технологии Corda таким образом, чтобы сопоставление заявок являлось умным контрактом, книга заявок хранилась у каждого участника биржи, не было возможности возникновения или исчезновения заявки из системы;
- протестировать данную систему.

Бакалаврская работа состоит из раздела «Обозначения и сокращения», введения, двух разделов основной части, заключения, списка использованных источников и восьми приложений. Общий объем работы — 82 страницы, из них 43 страницы — основное содержание, включая 8 рисунков. В списке ис-

пользованных источников содержится 21 наименование.

Первый раздел «Теоретические основы технологии блокчейн» содержит описание технологии блокчейн как структуры данных, особенностей децентрализации блокчейна, умных контрактов Smart contract, фреймворка Corda и его основных составляющих элементов.

Во втором разделе «Разработка и тестирование программного обеспечения» представлены проектирование и разработка приложения, основанного на технологии Corda для децентрализованной работы биржевой платформы, а также результаты тестирования разработанной платформы.

1 Теоретические основы технологии блокчейн

1.1 Блокчейн

1.2 Структура данных

Блокчейн — база данных с широкомасштабным тиражированием транзакций [2]. Данная база состоит из отдельных блоков, связанных друг с другом в виде непрерывной цепочки (рисунок 1). Каждый блок содержит в себе данные и хэш предыдущего блока, представляющий собой ссылку на предыдущий блок. Данный хэш позволяет поддерживать верный порядок блоков в цепочке.

Хэш блока зависит не только от порядка предыдущих блоков, но и от их содержимого. Это означает, что вносить изменения в уже существующий блок и удалять данные из существующего блока нельзя. Этим достигается уверенность в том, что с данные в блокчейне не будут подвержены изменениям. Также это свойство позволяет просмотреть всю цепочку транзакций и однозначно восстановить цепь событий в системе.

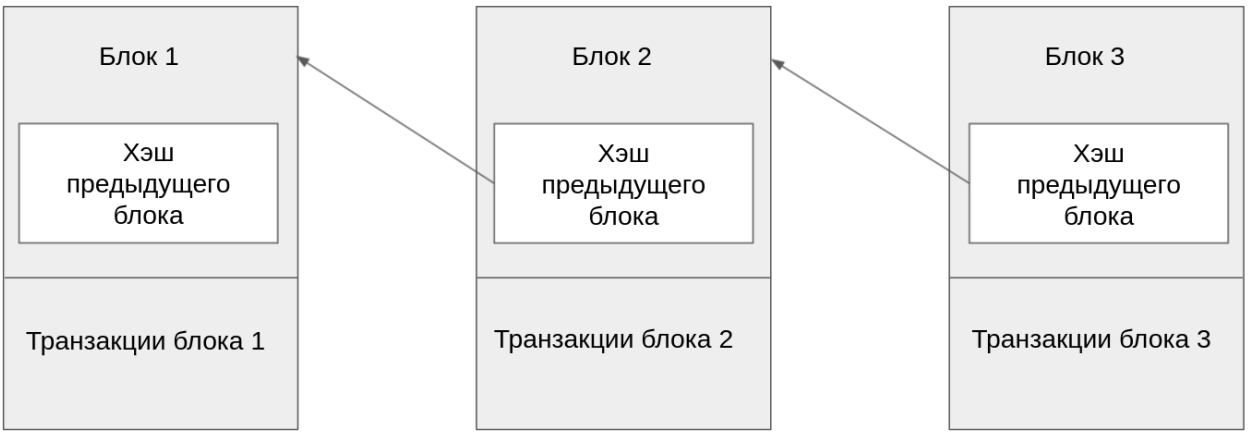

Рисунок 1 – Структура блокчейна

1.3 Децентрализованность

В блокчейне каждый узел сети имеет полную историю информации, которая была записана в цепочку блоков с момента её возникновения. Если один узел имеет ошибку в своих данных, то он может использовать остальные узлы как отправную точку для исправления своей информации. Таким образом, ни один узел в сети не может изменять информацию, хранящуюся в нем.

Если какой-то узел сети изменит информацию, то его будет очень легко вычислить, так как все узлы соединены друг с другом. Чтобы изменить способ

работы этой системы или информацию, хранящуюся в ней, большая часть вычислительной мощности децентрализованной сети должна будет согласовать указанные изменения. Это гарантирует, что любые происходящие изменения отвечают интересам большинства.

Таким образом, полный жизненный цикл транзакции состоит из следующих этапов:

1. Создание новой транзакции.
2. Передача транзакции всем узлам в сети.
3. Проверка каждым узлом сети валидности транзакции.
4. Сбор легитимных транзакций в блок транзакций.
5. Соединение блоков транзакций в единую цепочку, которая хранит всю историю транзакций и является постоянной.
6. Завершение транзакции.

1.4 Smart contract

Смарт-контракты — это приложения для автоматизации бизнеса, которые работают в децентрализованной сети, такой как блокчейн.

Фактически, смарт-контракт — это бизнес-правила, переведенные в программное обеспечение. Поскольку код смарт-контракта выполняется поверх открытой бухгалтерской книги блокчейна, правила могут применяться не только внутри компании, которая запрограммировала смарт-контракт, но и к другим бизнес-партнерам, которым разрешено находиться в блокчейне.

Еще одним преимуществом смарт-контрактов является скорость их выполнения. Как только условие выполнено, контракт немедленно исполняется. Поскольку смарт-контракты являются цифровыми и автоматизированными, нет необходимости обрабатывать документы и тратить время на согласование ошибок, возникающих из-за ручного заполнения документов [3].

1.5 Corda

Corda — фреймворк с открытым исходным кодом, представляющий собой распределенный реестр для хранения, управления и синхронизации финансовых обязательств между различными финансовыми организациями.

Corda состоит из следующих элементов [4] :

— сеть, в которой Corda существует;

— реестр и технология рапределения фактов между узлами;

— состояния — представляют общие факты в реестре;
— транзакции — обновляют состояния реестра;
— контракты — определяют способы, которыми состояния могут развиваться с течением времени;
— потоки — описывают взаимодействия, которые должны происходить между сторонами для достижения консенсуса (для удовлетворения бизнес-требований);
— узлы, каждый из которых содержит экземпляр Corda, одно или несколько приложений CorDapp.

1.5.1 Сеть

Сеть Corda — это одноранговая сеть узлов [5]. Каждый узел запускает программное обеспечение Corda (экземпляр Corda и одно или несколько приложений CorDapps [6]).

Все узлы в сети могут взаимодействовать друг с другом. Идентификатор узла используется для представления узла в транзакциях: например, если узел был вовлечен в транзакцию по покупке актива. Сервис карты сети сопоставляет каждый известный идентификатор узла с IP-адресом. Эти IP-адреса используются для обмена сообщениями между узлами.

1.5.2 Реестр

В Corda нет единого центрального хранилища данных. Вместо этого каждый узел ведет свою собственную базу данных тех фактов, о которых он знает [7].

Узел знает только о тех фактах, с которыми он связан. Стоит отметить, что, хотя в системе нет центрального хранилища, при желании можно транслировать основной факт всем участникам. Это можно сделать, перебрав все узлы из карты сети.

1.5.3 Состояния

Состояние — это неизменяемый объект, представляющий факт, известный одному или нескольким узлам Corda в определенный момент времени [8]. Состояния могут содержать произвольные данные, что позволяет им представлять факты любого рода.

Поскольку состояния являются неизменяемыми объектами, вместо изменения состояния жизненный цикл общего факта с течением времени представлен последовательностью состояний. Когда состояние нужно обновить, необходимо создать новую версию состояния, представляющую новое состояние, и пометить существующее состояние как историческое.

Такая последовательность замен состояний дает полное представление об эволюции общего факта с течением времени.

1.5.4 Транзакции

Corda использует модель UTXO (неизрасходованный вывод транзакции [9]), в которой каждое состояние в реестре неизменяемо. Реестр со временем развивается, применяя транзакции. Транзакции обновляют реестр, отмечая ноль или несколько существующих состояний реестра как исторические (входные данные) и создавая ноль или более новых состояний реестра (выходы).

Транзакции являются атомарными.

В случае цепочки транзакций при создании новой транзакции в выходных данных указывается, что предлагаемая транзакция еще не существует и, следовательно, должна быть создана предлагающим или предлагающими транзакцию. Однако, входные состояния уже существуют как выходы предыдущих транзакций. Поэтому необходимо включить их в предлагаемую сделку посредством ссылки.

1.5.5 Контракты

Транзакция действительна только в том случае, если она подписана цифровой подписью всех необходимых сторон. Кроме того, что транзакция должна иметь все необходимые подписи, она должна удовлетворять условиям контракта [10].

Действительность по контракту определяется следующим образом:

- каждое состояние транзакции определяет тип контракта;
- контракт принимает транзакцию в качестве входных данных и указывает, считается ли транзакция действительной в соответствии с правилами контракта;
- транзакция действительна только в том случае, если контракт каждого состояния ввода и каждого состояния вывода считает ее действительной.

1.5.6 Потоки

Сети Corda используют двухточечный обмен сообщениями вместо глобального вещания [11]. Это означает, что координация обновления реестра требует, чтобы участники сети точно указывали какая именно информация должна быть отправлена, каким контрагентам и в каком порядке.

Вместо того, чтобы указывать эти шаги вручную, Corda автоматизирует процесс с помощью потоков. Поток — это последовательность шагов, которая сообщает узлу как выполнить конкретное обновление реестра, например, выпустить актив или произвести расчеты по сделке.

После того как данный бизнес-процесс инкапсулирован в поток и установлен на узле как часть CorDapp, владелец узла может дать узлу команду запустить этот бизнес-процесс в любое время с помощью RPC вызова. Поток абстрагирует от владельца узла все проблемы, связанные с сетью, вводом-выводом и параллелизмом.

Вся деятельность на узле происходит в контексте этих потоков. В отличие от контрактов, потоки не выполняются в изолированной программной среде. Это означает, что узлы могут выполнять такие действия, как сетевое соединение, ввод-вывод при выполнении потока.

1.5.7 Узлы

Узел Corda — это среда выполнения JVM с уникальным идентификатором в сети, в которой размещены службы Corda и CorDapps [12].

Основными элементами архитектуры узла являются:

- база данных для хранения информации;
- сетевой интерфейс для взаимодействия с другими узлами;
- интерфейс RPC для взаимодействия с владельцем узла;
- интерфейс и поставщик CorDapp для расширения узла путем установки CorDapps.

Также в сети Corda обязательно должнен присутствовать узел Нотариус. Нотариальный узел обеспечивает консенсус в отношении уникальности, подтверждая, что для данной транзакции он еще не подписал другие транзакции, которые потребляют любое из входных состояний предложенной транзакции. Каждая транзакция должна пройти через нотариальный узел.

2 Разработка и тестирование программного обеспечения

Исходя из того, что Corda использует JVM [13], было принято решение разрабатывать систему на языке программирования Java.

2.1 Архитектура приложения

Разработанное приложение является узлом сети Corda, с которым можно взаимодействовать с помощью HTTP протокола. Приложение поддерживает создание новой биржевой заявки, отмену существующей биржевой заявки, запрос всех запросов на новую заявку.

В данном приложении умным контрактом является ядро биржевой платформы Matching engine. При успешном торге заявок в ядре биржевой платформы составляется блокчейн соглашение между участниками успешного сопоставления заявок. Инструмент, размер, цена соглашения при этом определяются данными, полученными из Matching engine.

Также приложение возвращает ответ на запрос всех соглашений, в которых принимает участие данный узел.

Веб-сервер для обеспечения работы по HTTP протоколу построен с применением технологии Spring.

2.2 Подключение необходимых библиотек

Для сборки проекта используется система автоматизированной сборки приложения Gradle. Кроме сборки приложения Gradle используется для подключения к проекту библиотек и фреймворков. Вместо того, чтобы доставлять нужные зависимости вручную, достаточно в специальном файле `build.gradle` описать необходимые зависимости в разделе `dependencies` с использованием ключевого слова `implementation` [14].

2.3 Создание классов состояний

Для корректной работы приложения состояния обязательно должны содержать следующие поля: отправитель состояния, получатель состояния, а так же уникальный идентификатор состояния. Кроме этого, так как состояния сохраняются в базу данных, был определен вспомогательный класс для корректной работы приложения с базой данных.

Так как приложение поддерживает постановку новой заявки, отмену старой, а так же составление блокчейн соглашения, было создано 3 класса состо-

яний: NewOrderState, CancelOrderState, IOUState.

2.4 Создание классов контрактов

Так как приложение поддерживает постановку новой заявки, отмену старой, а так же составление блокчейн соглашения, было создано 3 класса контрактов: NewOrderContract, CancelOrderContract, IOUContract. Контракты проверяют валидность данных, содержащихся в транзакции.

2.5 Создание классов потоков

Потоки делятся на 2 типа: потоки-инициаторы и инициируемые потоки. Поток-инициатор порождает транзакцию, а инициируемый поток принимает транзакцию, должен ее подписать и отправить обратно.

Было создано 6 классов потоков: IOUInitiator, NewOrderInitiator, CancelOrderInitiator, IOUResponder, NewOrderResponder, CancelOrderResponder.

2.6 Построение веб сервиса

Для взаимодействия с узлом сети необходимо подключиться к нему с помощью RPC подключения. Для построения веб-сервера создан контроллер Controler, который отвечает за прием HTTP запросов.

Принятие новой заявки будет осуществляется с помощью POST запроса по адресу neworder, телом которого является объект класса NewOrderRequest. Далее всем узлам сети будет отправлена новая транзакция с сообщением о постановке новой биржевой заявки.

Принятие запроса на отмену заявки также будет осуществляться с помощью POST запроса по адресу cancelorder , но телом его будет являться объект класса CancelOrderRequest. Далее необходимо всем узлам сети будет отправлена новая транзакция с сообщением об отмене биржевой заявки.

Для получения информации о находящихся состояниях в реестре разработаны следующие методы:

— Метод, принимающий GET запрос по адресу ious, возвращает все блокчейн соглашения о передаче инвестиционных инструментов, содержащиеся в реестре.
— Метод, принимающий GET запрос по адресу neworders, возвращает все запросы на постановку новой заявки в книгу заявок.

— Метод, принимающий `GET` запрос по адресу `cancelorders`, возвращает все запросы на отмену заявки.
— Метод, принимающий `GET` запрос по адресу `myious`, возвращает блокчейн соглашения о передаче инвестиционных инструментов, в которых текущий узел сети является принимающей стороной.

ЗАКЛЮЧЕНИЕ

В ходе выполнения работы был получен опыт разработки приложения с использованием блокчейн технологии Corda.

Реализовано приложение, которое является децентрализованной биржевой платформой. Также реализован веб сервис для взаимодействия с данным приложением с использованием технологии Spring Boot.

Продемонстрирована работа приложения с успешно отправленными запросами, полученными ответами, а так же с демонстрацией работы технологии Smart contract.

Концепция блокчейна может положить конец нынешней иерархии власти в информационном поле [15]. В результате способность накапливать и обрабатывать данные перейдет к децентрализованной, никому не принадлежащей структуре.

Важно понимать, что блокчейн является архитектурой, на основе которой могут быть построены различные приложения. В данной работе рассмотрено применение блокчейна в секторе финансовых технологий. Также блокчейн может быть применен для реализации различных целей. С помощью блокчейна можно производить документооборот компании, управлять цепями поставок.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 *Лобанова, Ю. К.* Перспективы и применение технологии блокчейн в современном мире // ЭКОНОМИКА ГЛАЗАМИ МОЛОДЫХ. Материалы Региональной научно-практической конференции студентов и молодых ученых. — Vol. 1. — Томск, Россия: 2019. — Pp. 204–207.

2 *Равал, С.* Децентрализованные приложения. Технология Blockchain в действии. / С. Равал. — Санкт-Петербург: Питер, 2017. — 240 с.

3 What are smart contracts on blockchain | IBM [Электронный ресурс]. — URL: https://www.ibm.com/topics/smart-contracts (Дата обращения 20.05.2021). Загл. с экр. Яз. англ.

4 Key concepts | Corda OS 4.8 | Corda Documentation [Электронный ресурс]. — URL: https://docs.corda.net/docs/corda-os/4.8/key-concepts.html (Дата обращения 06.05.2021). Загл. с экр. Яз. англ.

5 Network | Corda OS 4.8 | Corda Documentation [Электронный ресурс]. — URL: https://docs.corda.net/docs/corda-os/4.8/key-concepts-ecosystem.html (Дата обращения 10.05.2021). Загл. с экр. Яз. англ.

6 What is a CorDapp? | Corda OS 4.7 | Corda Documentation [Электронный ресурс]. — URL: https://docs.corda.net/docs/corda-os/4.7/cordapp-overview.html (Дата обращения 01.05.2021). Загл. с экр. Яз. англ.

7 Ledger | Corda OS 4.8 | Corda Documentation [Электронный ресурс]. — URL: https://docs.corda.net/docs/corda-os/4.8/key-concepts-ledger.html (Дата обращения 09.05.2021). Загл. с экр. Яз. англ.

8 States | Corda OS 4.8 | Corda Documentation [Электронный ресурс]. — URL: https://docs.corda.net/docs/corda-os/4.8/key-concepts-states.html (Дата обращения 07.05.2021). Загл. с экр. Яз. англ.

9 Что такое UTXO в сети биткоин — объясняем простыми словами. Обзор 2021 - Coin Post [Электронный ресурс]. — URL: https://coinpost.ru/

p/chto-takoe-utxo-v-seti-bitkoin-obuyasnyaem-prostymi-slovami (Дата обращения 01.05.2021). Загл. с экр. Яз. рус.

10 Contracts | Corda OS 4.8 | Corda Documentation [Электронный ресурс]. — URL: https://docs.corda.net/docs/corda-os/4.8/key-concepts-contracts.html (Дата обращения 09.05.2021). Загл. с экр. Яз. англ.

11 Flows | Corda OS 4.8 | Corda Documentation [Электронный ресурс]. — URL: https://docs.corda.net/docs/corda-os/4.8/key-concepts-flows.html (Дата обращения 15.05.2021). Загл. с экр. Яз. англ.

12 Nodes | Corda OS 4.8 | Corda Documentation [Электронный ресурс]. — URL: https://docs.corda.net/docs/corda-os/4.8/key-concepts-node.html (Дата обращения 11.05.2021). Загл. с экр. Яз. англ.

13 Что такое JVM? Знакомство с виртуальной машиной Java [Электронный ресурс]. — URL: https://ru.hexlet.io/blog/posts/chto-takoe-jvm-znakomstvo-s-virtualnoy-mashinoy-java (Дата обращения 05.05.2021). Загл. с экр. Яз. рус.

14 Build Script Basics [Электронный ресурс]. — URL: https://docs.gradle.org/current/userguide/tutorial_using_tasks.html (Дата обращения 19.04.2021). Загл. с экр. Яз. англ.

15 *Винья, П.* Машина правды. Блокчейн и будущее человечества. / П. Винья, М. Кейси. — Москва: Манн, Иванов и Фербер, 2018. — 320 с.