ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»
(Н И У « Б е л Г У »)

ИНСТИТУТ ИНЖЕНЕРНЫХ И ЦИФРОВЫХ ТЕХНОЛОГИЙ
КАФЕДРА ПРИКЛАДНОЙ ИНФОРМАТИКИ И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

РАЗРАБОТКА ТОРГОВОГО СОВЕТНИКА В СРЕДЕ METATRADER ДЛЯ ТОРГОВЛИ НА ФИНАНСОВЫХ РЫНКАХ

Выпускная квалификационная работа
обучающегося по направлению подготовки 09.03.03
Прикладная информатика
очной формы обучения, группы 12001504
Богомазова Александра Михайловича

Научный руководитель
Старший преподаватель
Болгова Е.В.

БЕЛГОРОД 2019

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Со времен появления компьютеров популярность среди людей, торгующих на биржах, набирали автоматизированные торговые системы. Одним из популярных методов торговли на бирже через Интернет является интернет-трейдинг. Популярность этого подхода к торговле растет с каждым годом, но все аспекты построения эффективной торговой системы, даже сегодня, до конца не изучены.

С появлением интернет-трейдинга не только возросла скорость торговых операций, но и появились программы для полной или частичной автоматизации деятельности трейдеров.

Из-за больших объемов операций с иностранной валютой на бирже инвесторам трудно обрабатывать объемное количество данных, следовательно, большинство из них не получают максимальной прибыли. Однако невозможно оперативно получить необходимые математические расчеты без использования специализированного программного обеспечения, а индикаторы позволяют совершать математические операции своевременно без вмешательства человека. Чтобы максимизировать прибыль в торговой стратегии, необходимо использовать индикаторы, которые рассчитываются на основе исторических данных торгового инструмента. Поэтому задача разработки набора индикаторов является актуальной.

Объектом исследования данной работы является финансовый рынок.

Предметом исследования является процесс торговли на финансовом рынке.

Целью работы является повышение финансовой доходности торговой стратегии.

Для достижения цели необходимы выполнить некоторые задачи:

- проанализировать предметную область и выявить недостатки в действующей торговой системе;

- выполнить разработку алгоритмов работы индикаторов;

- выполнить разработку набора индикаторов;

- протестировать индикаторы и торговую стратегию с использованием индикаторов;

- проанализировать экономическую эффективность улучшенной торговой стратегии.

Пояснительная записка к выпускной квалификационной работе состоит из: введения, трех разделов, заключения, списка библиографических источников, приложения.

В первом разделе выполняется анализ предметной области, постановка задачи, выявление недостатков торговой стратегии.

В втором разделе осуществляется проектирование разрабатываемых индикаторов, построение модели «как должно быть», способы модифицирования финансовой торговой стратегии.

В третьем разделе описывается реализация программного обеспечения с описанием используемых средств, этапов разработки и оценка экономической эффективности улучшенной торговой стратегии.

Данная работа изложена на 50 страницах и имеет 31 рисунок, 2 таблицы и 2 приложения.

1 **Аналитическая часть**

1.1 Технико-экономическая характеристика предметной области

Финансовый рынок – это структура, которая позволяет в рыночной экономике покупать и продавать ценные бумаги, инвестиционные товары, например, драгоценные металлы. Оборот финансовых рынков может также включать другие активы с высокой ликвидностью.

Биржа – организация, обеспечивающая и регулирующая торговлю ценными бумагами, валютами, другими финансовыми инструментами и их производными [1-3].

Индикаторы – интерпретация ряда тех или иных рыночных параметров наглядно показывающая, куда предположительно будет двигаться цена рассматриваемого финансового инструмента [1-3].

Торговая система – порядок действий трейдера, основанный на комплексном анализе рынка и, в соответствии с вытекающими из этого, анализа сигналами. В качестве сигналов могут служить показания различных индикаторов, пробои различных уровней и т.п. [4-6].

Электронная торговля всегда осуществляется в одном и том же месте, в строго определенный интервал времени – во время биржевой сессии и в соответствии с четко установленными, обязательными для всех участников правилами.

Для международного характера товарных бирж требуется большой объем операций с соответствующими товарами, отвечающий потребностям всего мирового рынка, а также предоставление свободных валютных, торговых и налоговых режимов, что способствует участию иностранных участников в биржевой торговле. Они проводят операции, ориентированные на внутренний рынок, часто с ограничениями в торговых и налоговых режимах [7,8].

Финансовый советник следит за изменениями котировок на финансовом рынке на основе критериев, изложенных в нем. Он также может действовать как система с сигналами, и человек решает, открывать сделку или нет. У торгового советника есть ряд положительны факторов:

- отсутствие эмоций. Торговый советник следует неукоснительно и безусловно по указанному алгоритму, не допуская отклонений от него. Следуя поставленной задаче, он позволит избавиться от несистемных транзакций.

- дисциплина. Дисциплина сохраняется даже при нестабильной ситуации на финансовом рынке. Инвесторы периодически совершают ошибки из –за боязни понести убытки или из-за жадности в течение длительного.

- последовательность. Большой проблемой при торговле – выбор между планированием торговли и торговлей согласно плану. Даже когда трейдер знает, что его система прибыльна в долгосрочной перспективе, краткосрочная серия убыточных сделок может заставить его отклониться от плана, тем самым «убив» все ожидания от стратегии. Торговые стратегии обеспечивают четкую последовательность действий и исключают возможность ошибок при вводе данных при открытии позиции. [9-11]

- скорость. Поскольку компьютер позволяет выполнять анализ и расчеты с высокой скоростью, тем самым практически мгновенно реагируя на изменения рыночных условий, торговая система немедленно открывает позицию для одного или даже нескольких торговых инструментов. Отклонение на несколько секунд от торгового плана может сильно изменить результаты.

- диверсификация. Торговые стратегии позволяют трейдеру вести торговлю одновременно на нескольких счетах или инструментах, что позволяет распределить риски. Что кажется невозможным для человека, торговый советник сделает это за считанные секунды. Система позволяет отслеживать единовременно несколько рынков и контролировать множество позиций.

Торговый советник имеет достаточно большое количество плюсов, но не исключает минусы. Ниже приведены недостатки, на которые стоит обратить внимание при использовании торговых советников:

- недостаток технологичности. Теоретически торговые советники работают тривиально: устанавливается программа и правила, после чего ведется торговля. Однако в действительности это сложный процесс, основанный на многоуровневых технологиях. В зависимости от торговой платформы позиция может храниться на компьютере, а не на сервере. Если связь с Интернетом потеряна, позиция не будет отправлена на рынок, что может привести к большим потерям.

- контроль. Могут возникнуть технические проблемы (поломка компьютера, проблемы с подключением, сбой питания или другие сбои системы). Могут быть внутренние проблемы с советником, непреднамеренная отправка позиции или двойные объемы. Если система контролируется, такие сбои легко устраняются.

- переоптимизация. Частой проблемой является повторная оптимизация советника. Трейдеры, которые выбирают точные параметры торгового советника на основе исторических данных после тестирования, на реальных данных могут понести огромные убытки.

- фундаментальный анализ. Если провести технический анализ для торгового советника не составит труда, так как технические индикаторы – это набор формул, то фундаментальный анализ и трейдинг на основе новостей для него задача сложная.

- изменчивость. Советник может быть очень хорошим при определенных условиях рынка, но если они начнут меняться, то и эффективность вместе с ними. Не существует универсального торгового советника, одинаково хорошо работающего при любых условиях рынка, поэтому со временем необходимо вносить изменения в его работу.

Автоматизированные торговые системы или торговые советники собирают массивные объемы информации на финансовых рынках каждый год.

Но, прежде чем учиться их создавать, нужно достаточно хорошо разбираться в биржевой торговле и финансовых рынках.

1.2 Обоснование необходимости и цели использования вычислительной техники для решения задачи

На данный момент более 84% всех транзакций на финансовом рынке совершаются на электронных платформах с использованием специализированных терминалов для торговли.

Высокочастотный трейдинг – это торговля, когда сделки по покупке и продаже ценных бумаг открываются и закрываются в течение нескольких секунд. Для такой торговли используются торговые советники, а компании располагают офисы максимально приближенно к серверам, чтобы уменьшить погрешности исполнения.

Из-за этого инвесторы теряют прибыль, а прибыль является основной целью. Проведя анализ такого типа торговли можно сделать вывод, что использование компьютера со специализированным программным обеспечением необходимо для прибыльной торговли. [9-11]

После улучшения условий для торговли требуемая скорость не достигается, поскольку расчеты индикаторов необходимо проводить вручную. Чтобы решить эту проблему, необходимо использовать торговые терминалы. Сделки все равно будут выполняться вручную. Наиболее подходящий программный продукт – MetaTrader4.

Для разрабатываемых моделей использовалась методология IDEF0. Эта методология отображает иерархическую модель системы диаграмм и описание частей системы.

На рисунке 1.1 показана контекстная диаграмма, отображающая процесс осуществления торговли, а также входящая и выходящая информация,

управление, механизмы. Диаграмма на рисунке 1 демонстрирует ситуацию – «КАК ЕСТЬ».

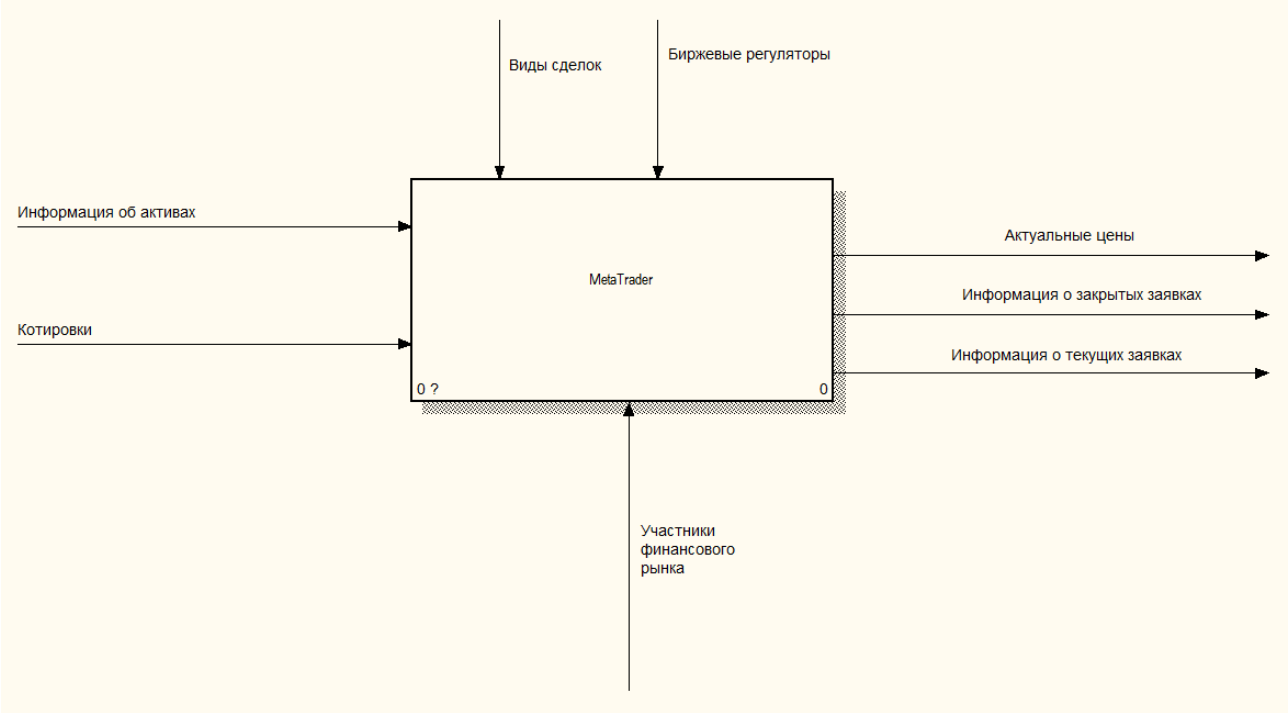


Рисунок 1.1 – Контекстная диаграмма

Как показано на рисунке 1.1 MetaTrader принимает входящую информацию в форме «Котировки» и «Информацию об активах». Эта информация поступает от брокера, механизмом являются участники финансового рынка, осуществляющие там торговую деятельность. Управлением на данной диаграмме являются правила торговли на финансовых рынках и «Биржевые регуляторы», которые осуществляют контроль за ходом торгов исключая мошенничество.

На рисунке 1.2 представлена декомпозиция контекстной диаграммы, показывающая процесс прохождения информации в MetaTrader.

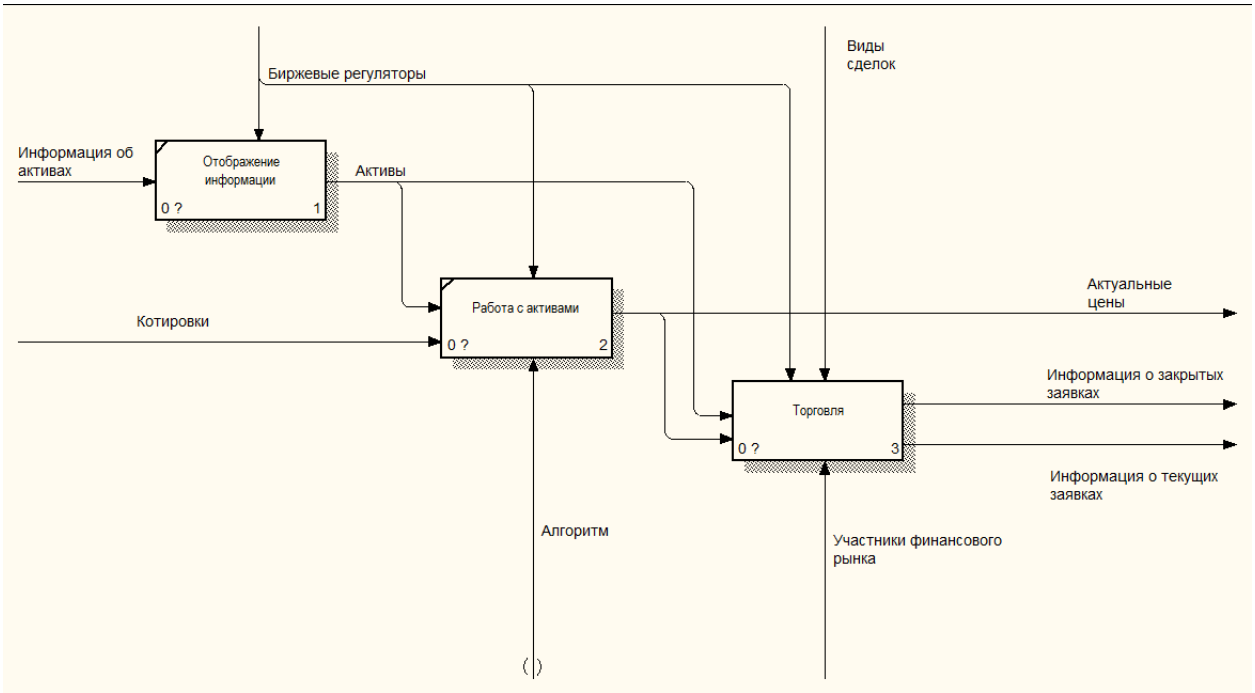


Рисунок 1.2 – Декомпозиция контекстной диаграммы

Процесс отображения информации обрабатывает попадающую в него «Информацию об активах» и выводит её экран, после чего обработанные активы отправляются в процесс «Работы с активами». В процессе «Работа с активами» алгоритм программного продукта MetaTrader после принятия котировки проверяет наличие ошибок и отображает текущие цены по активу пользователю. Процесс «Торговли» является более сложным, так как в нем принимают участие пользователи информационной системы. Декомпозиция процесса «Торговли» показана на рисунке 1.3.

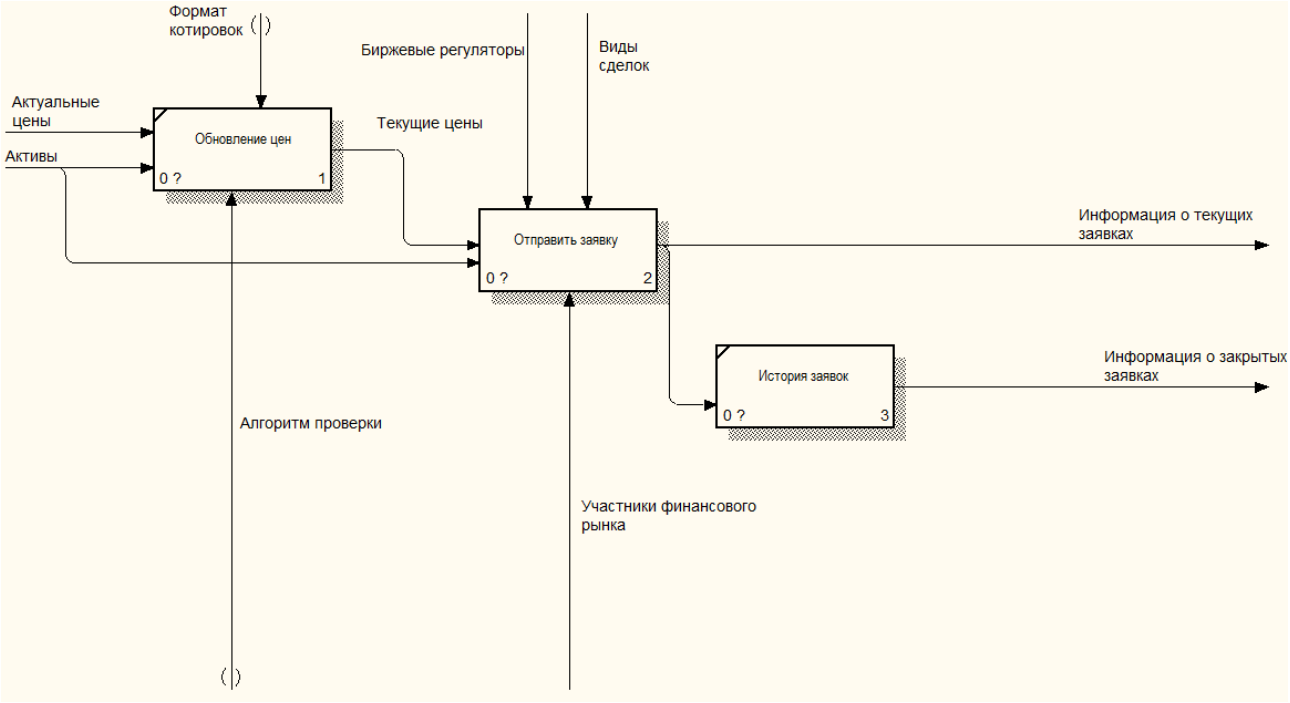


Рисунок 1.3 – Декомпозиция процесса «Торговля»

Для того чтобы не совершались неразрешенные изменения в котировках, MetaTrader проводит проверку в процессе обновление цен. Туннель «Формат котировок» указывается в настройках MetaTrader пользователем. После обновления цены данные переходят в процесс «Отправить заявку». Процесс «Отправить заявку» необходимо рассмотреть более подробно, декомпозиция этого процесса представлена на рисунке 1.4. Когда сделка завершена, она записывается в историю заявок и отправляется брокеру в одном формате данных, а пользователю в другом.

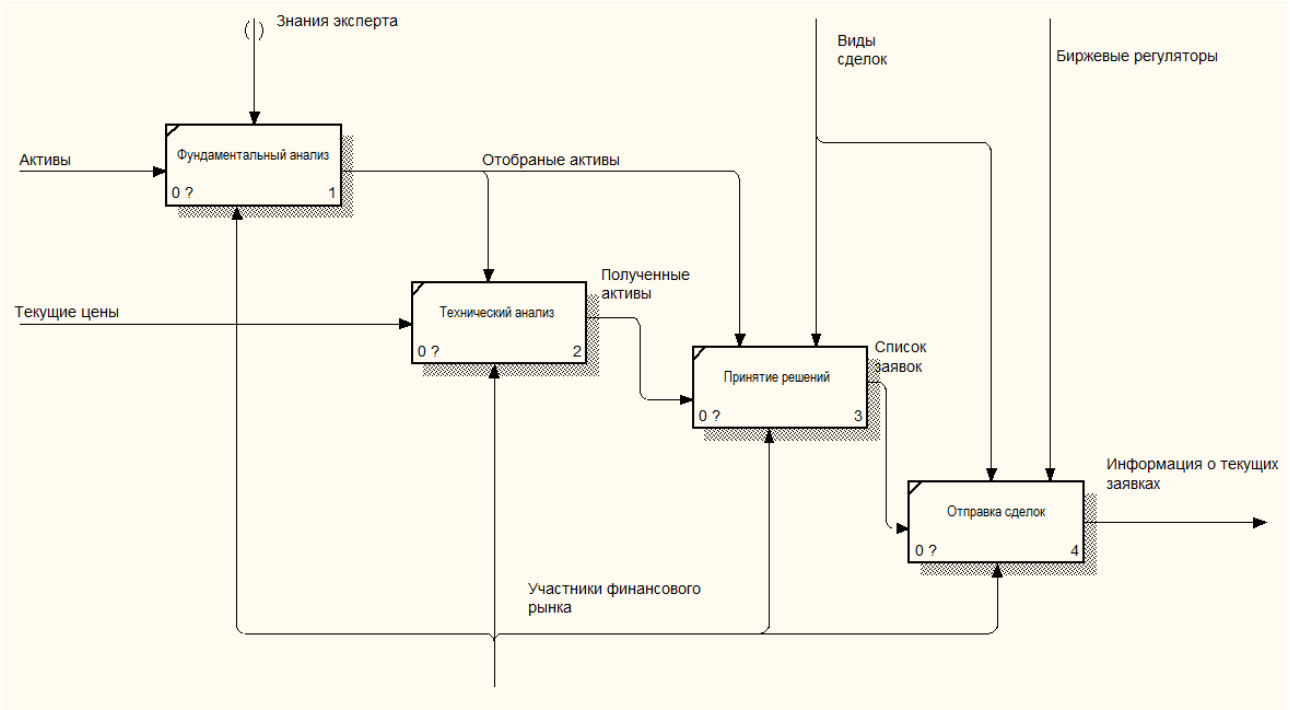


Рисунок 1.4 – Декомпозиция процесса «Отправить заявку»

На рисунке 1.4 показан этап, на котором происходит процесс выполнения сделки участником на финансовом рынке. Проведя анализ данного этапа можно заметить не эффективность данного решения. В ходе фундаментального анализа инвестор, имея информацию об активах, оценивает актив на возможность дальнейшего роста используя различную информацию, такую как:

- выплата дивидендов;

- последние новости, связанные с активом;

- финансовые показатели (чистая прибыль, рентабельность, чистая стоимость компании, движение денежных средств и т.д.).

Данные показали помогают отбросить заранее убыточные активы. После чего обработанная информация переходит в технический анализ, в котором на основе текущих и исторических цен, проводится расчет индикаторов. Инвестор на основе своих соображений проводит отбор индикаторов, установку параметров и после проверяет на придельные значения. Тем самым проводя очередную фильтрацию активов. После результаты переходят в процесс «Принятия решений», где участник финансового рынка проводит

выбор, на основе своего баланса и более приоритетных активов (у какого значения было более точным в фундаментальном и техническом анализе) по каким активам открывать позиции. После завершения такого сложного и длительного процесса список заявок на открытие позиций отправляется брокеру, если все условия, такие как правильные данные и соответствие баланса пользователя, были соблюдены, сделки будут успешно открыты. В противном случае пользователь получит сообщение о ошибке.

Графическое отображение процессов показывает, что программное обеспечение, позволяет участнику торгового рынка более быстро принимать решение, так как отпадает необходимость в расчетах значений индикаторов самому.

Так как можно использовать торговых советников, которые будут производить операции строго по алгоритму, не совершая отклонений от торговой стратегии, что позволяет исключить любой человеческий фактор (страх потери, желание получения большей прибыли, ошибки при расчетах или при отправке заявки). Торговый советник будет сам проводить как фундаментальный, так и технический анализ, чем исключит вероятность ошибки в расчетах. Инвестору необходимо указать параметры для отбора активов, по которым должен действовать торговый советник.

Также инвестор с помощью торгового советника имеет возможность протестировать свою стратегию на исторических данных. Эта возможность позволят оценить и отредактировать любую торговую идею и определить возможные прибыли или убытки, тем самым, не допустив потерь на реальном счете.

Вышеописанный процесс позволяет сделать вывод в необходимости использования прикладных решений в виде терминала, а также торговых советников и индикаторы для прибыльной торговли.

1.3 Постановка задачи

Набор индикаторов, разрабатываемый для торговли, представляет собой технологическую (формальную) систему, а разрабатываемая торговая система систему управления.

Технологическая система (формальная) – совокупность операций в процессе достижения цели. Структура такой системы определяется совокупностью методик и регламентов, а также правил и норм. Под процессом понимаются последовательные действия по изменению объекта.

При разработке индикаторов, чтобы получить требуемые значения в определенном диапазоне необходима последовательность действий над котировками.

Реализация индикаторов должна обеспечивать следующие функциональные возможности:

- своевременно производить расчеты без изменения сроков котировок;
- точные расчеты независимо от количества пунктов в цене;
- обеспечить доступ к расчетам индикатора от торгового советника;
- возможность настройки любого из параметров.

Система управления, обеспечивающая реализацию заданных целей, рассматривается как действие или функция и содержит два главных элемента:

- управляемая подсистема (объект управления);
- управляющая подсистема (выполняющая функцию управления).

Управляемая подсистема – торговый счет участника торгов, а управляющая подсистема – MetaTrader.

Торговый советник должен обеспечить реализацию своих функциональных возможностей:

- возможность настройки любых предельных значений индикатора для проведения сделок;
- проведение торговых операций на финансовом рынке;

- безотказная работа алгоритма;
- расчет финансовых показателей при тестировании для корректировок.

1.4 Анализ существующих торговых терминалов

Так как для реализации торгового советника можно использовать различные инструменты, следует проанализировать существующие аналоги.

NinjaTrader – признанная профессионалами торговая платформа с огромным функционалом. Торговая платформа NinjaTrader, имеет все возможности, что и другие платформы, она может создавать торговые стратегии, проводить тестирование их в реальном времени и вести процесс торговли в автоматическом режиме. NinjaTrade имеет более сотни индикаторов, разнообразные интервалы и высокая визуализация графиков. Она имеет возможность Демо – режима, а главное, все вышеперечисленное, дает возможность подстроить систему под любого трейдера, под желаемый стиль торговли. Но все обновления как бы не повышали функционал и возможности этой платформы, выйти беспрекословные лидеры среди платформ у NinjaTrader не выходит. [9-11]

Не смотря на все возможности есть ряд недостатков, которые срабатывают как отталкивающий фактор. Отсутствие серьезной разницы между базовой и расширенной версией. Должна быть единая версия с наилучшей возможностью адаптации к требованиям трейдера. Как правило, платформу предоставляет брокер, который заинтересован в получении трейдером дохода. Больше доход – больше депозит, что означает больше лот и в следствии увеличении дохода брокера. Если брокер, у которого открыт депозит, предлагает купить вероятно, у него проблемы с финансами. В современное время, когда большинство людей имеют доступ к интернету, а, следовательно, и к большим объёмам информации, это просто недопустимо

[11-13]. В настоящее время существует много бесплатных и эффективных программ. Так как есть большой выбор торговых платформ, фактор покупки расширенной версии негативно влияет использование данной платформы участниками рынка. Пример открытия сделки в терминале NinjaTrade изображен на рисунке 1.5.

Рисунок 1.5 – Торговый терминал NinjaTrader

MetaTrader 4 – одна из самых распространенных и известных платформ для торговли на финансовых рынках, в большей степени на рынке валют. Данная торговая платформа MT4 является разработкой компании «MetaQutes Software Corp», публичный выпуск был произведен 1 июля 2005 года, также была выпущена пятая версия программы, которая и должна была заменить MT4, но до сих пор плохо оптимизирована для рядового пользователя. Торговая платформа MT4 имеет множество достоинств, к важнейшим из них можно отнести:

- удобный интерфейс;
- возможность использования для доступа мобильные устройства;
- ведение нескольких счетов;
- интегрированные графические и технические инструменты;
- автоматический трейдинг.

Преимуществ у платформы достаточно, разработчики стараются делать все возможное, чтобы торговля через MT4 была доступна и комфортна для

каждого желающего. MetaTrader4 является многоязычной платформой, через неё проводят свои торговые операции несколько миллионов трейдеров по всему миру. Большую долю популярности платформе приносит возможность использования торговых советников – автоматизированных систем. При таком виде торговли всем процессом, начиная от заявки и до закрытия сделки, управляет робот. Большинство форекс – брокеров или дилинговые центры, которые предоставляют услуги по торговле валютой, CFD и фьючерсами используют торговую платформу MT4, что она, по сути, становится монополистом в данном секторе услуг [13-16]. Пример окна торгового терминала изображен на рисунке 1.6.

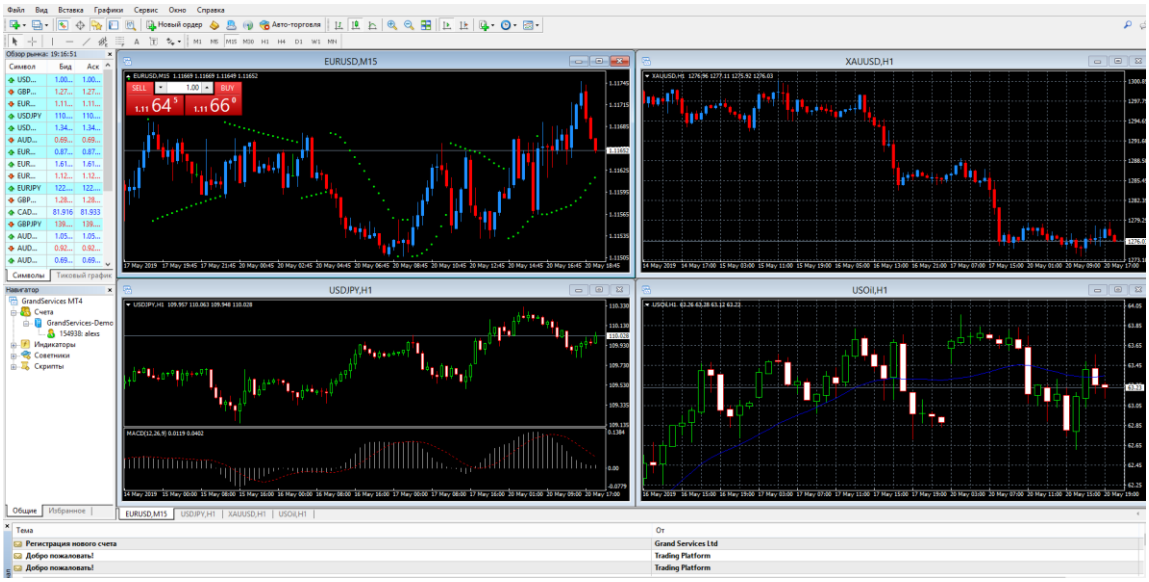


Рисунок 1.6 – Терминал MetaTrader 4

MetaTrader 5 – платформа для торговли, позволяющая трейдеру участвовать в торгах на биржах, совершать покупку фьючерсов и заключать контракты на разницу. Пользователь MetaTrader 5 может вести торговлю сразу на нескольких биржах и финансовых рынках. Разработчики придерживались концепции «все в одном месте», то есть постарались вместить в нее все, что только могло понадобиться трейдеру для успешной торговли – технический анализ, систему для тестирования торговых систем, автоматическую торговлю и многое другое. Вот основные особенности MetaTrader 5. [17,18]

Важнейшей составляющей прибыльной деятельности трейдера является совершения аналитических действий относительно дальнейшего движения цен различных финансовых инструментов. На рисунке 1.7 показано окно терминала.

Рисунок 1.7 – Терминал MetaTrader 5

QUIK – это специализированная программа для работы с биржевой информацией. Для передачи данных между терминалом пользователя и сервером QUIK.

Программное обеспечение Quik используется для организации доступа к биржевым рынкам. Quik, как программа интернет – торговли, имеет большую известность среди трейдеров. Quik, благодаря большому количеству функциональных возможностей, содержит инструменты, которые позволяют реализовывать торговые алгоритмы. Одним из них является язык Qpile. Quik. Qpile обладает маленьким набором возможностей, в сравнении с языками высокого уровня, как C++ или C#, но предоставляемых возможностей языка хватает для создания простых стратегий. Основные преимущества торгового терминал QUIK:

- высокая скорость получения информации и выполнения заявок;

- оптимизированный протокол передачи данных;

- использование «крепких» средств защиты информации;

- поддержка торговых операций на биржевых площадках;

- неплохо развитый функционал работы с заявками;

- встроенный язык QPILE для реализации таблиц с расчетными параметрами.

Пример окна QUIK изображен на рисунке 1.8.

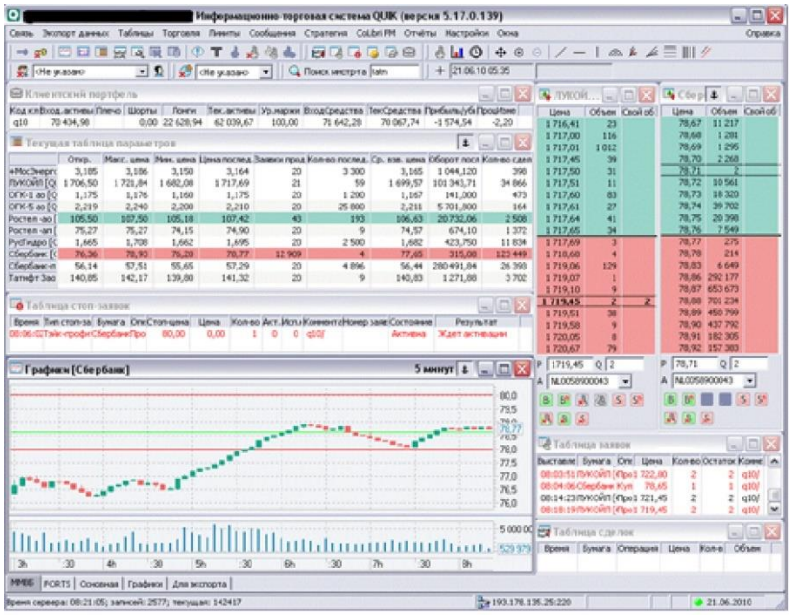


Рисунок 1.8 – Торговый терминал QUIK

StockSharp предназначен для автоматизации биржевой торговли. В состав S# входят следующие бесплатные программы такие как: S#.API, S#.Designer, S#.Data. API библиотека для профессиональной разработки торговых советников на языке высокого уровня C#. Эта библиотека позволяет реализовывать стратегии любой сложности. Её специфическими свойствами является: высокая отказоустойчивость, мобильность, скорость и производительность, тестирование на реальных данных (тестирование на исторических данных с точностью стремящийся к максимальной). [19-22]

На рисунке 1.9 показан терминал S#.

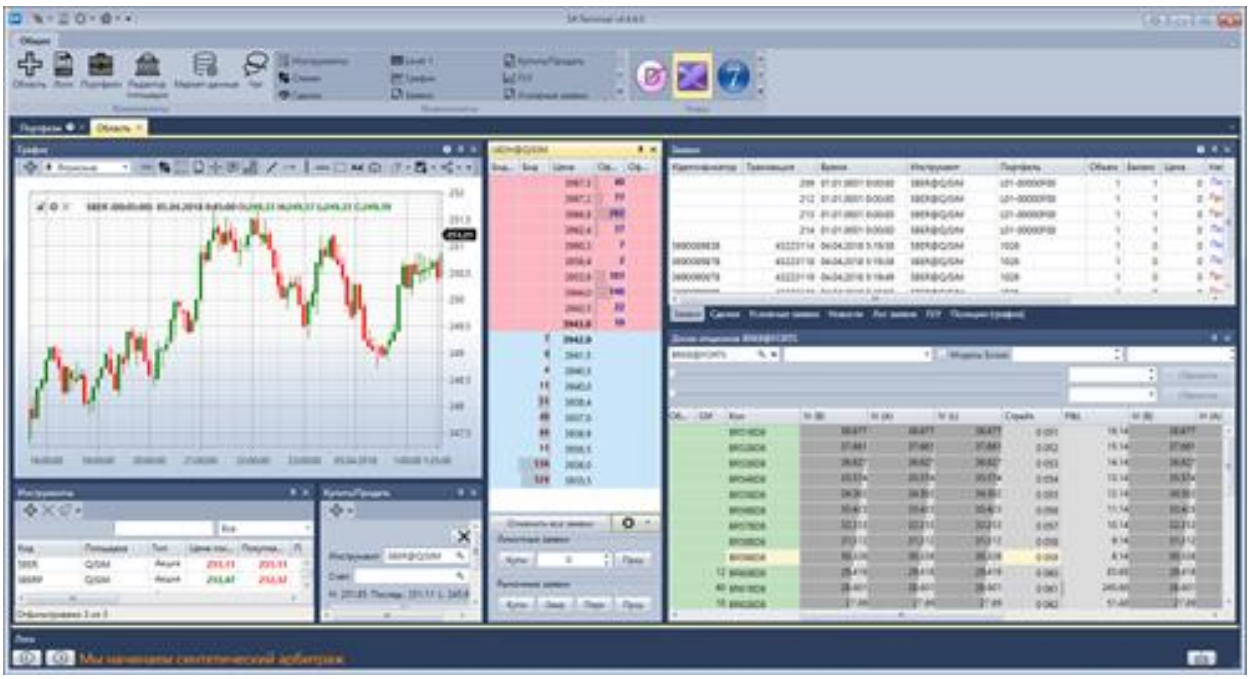


Рисунок 1.9 – Торговый терминал S#

В данном разделе был проведен анализ предметной области, также были рассмотрены существующие торговые терминалы для ведения торгов трейдерами

Таким образом, был завершен первый этап написания выпускной квалификационной работы и собраны основные необходимые сведения.

2 **Проектная часть**

2.1 Обоснование проектных решений

2.1.1 **Обоснование по программному обеспечению**

Имеется немалое количество информационных систем, которые предоставляют доступ финансовому рынку на бирже. В данном разделе необходимо провести анализ и выбрать более подходящую систему.

Торговый терминал – это программный комплекс, с помощью которого брокерская компания предоставляет возможность трейдеру получить доступ к рынку. Торговля на валютном рынке стала простой и доступной благодаря терминалу и сети Интернет.

Поэтому доступ к внешним информационным системам предоставляется, в зависимости от требуемых функций функций бесплатно или платно.

Перед выбором информационной системы, необходимо произвести анализ и выделить основные преимущества каждой из систем и сделать выбор по необходимым признакам.

Признаки, которым должен соответствовать торговый терминал для возможности осуществления всех возможностей торговой стратегии:

- доступность, клиент должен иметь постоянный доступ к торговому терминалу без каких –либо денежных оплат;

- популярность, торговый терминал должен быть известным среди брокеров, чтобы в случае смены брокера или других обстоятельств доступ к торговой стратегии оставался;

- информативность, торговый терминал должен быть информативным и иметь справочник на доступном языке;

- программируемость, терминал должен иметь возможность внесения изменений, создания торговых советников и индикаторов;

- производительность, терминал должен быть эффективным по техническим требованиям.

Далее необходимо провести оценку по требуемым критериям. Для выбора торгового терминала был использован метод анализа иерархий. На рисунке 2.1 представлены созданные узлы для построения иерархии.

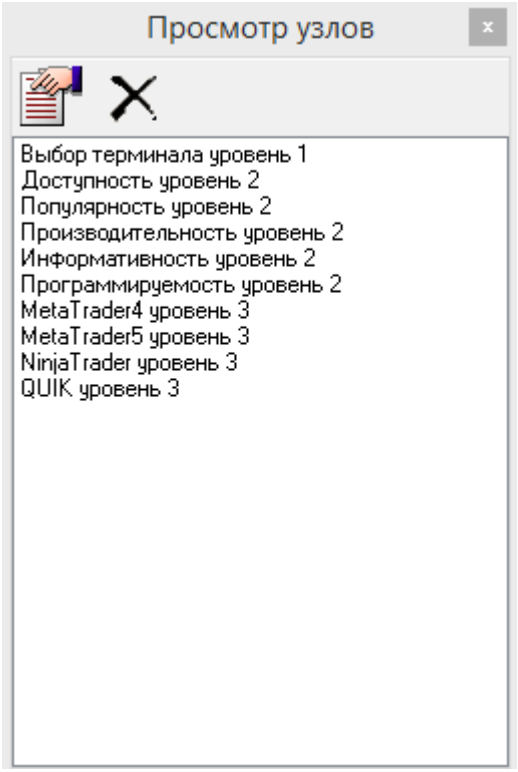
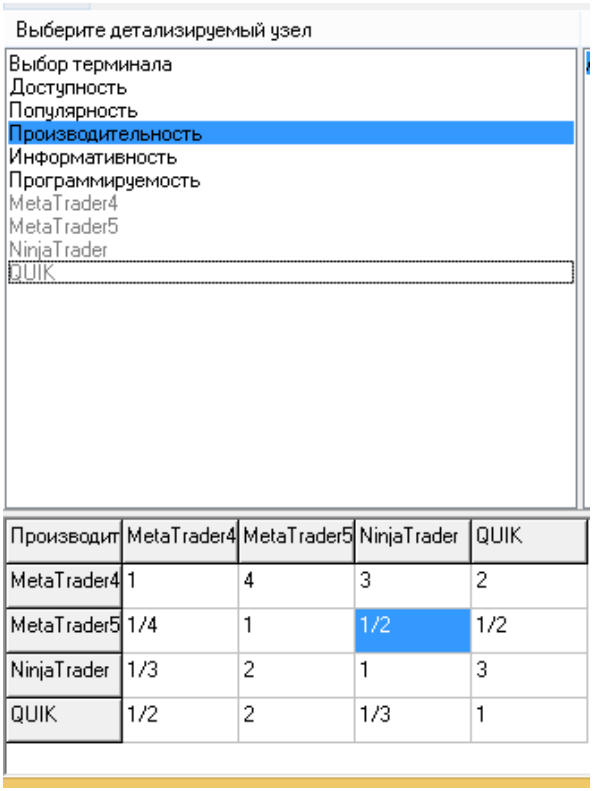


Рисунок 2.1 – Просмотр узлов

После создания узлов и построения иерархии требуется провести заполнение парных сравнений в блоке сравнительных суждений. На рисунке 2.2 показан пример заполнения.



Производит	MetaTrader4	MetaTrader5	NinjaTrader	QUIK
MetaTrader4	1	4	3	2
MetaTrader5	1/4	1	1/2	1/2
NinjaTrader	1/3	2	1	3
QUIK	1/2	2	1/3	1

Рисунок 2.2 – Заполнение сравнений

С помощью блока синтеза был получен результат, показывающий подходящий торговый терминал. Блок синтеза с результатом представлен на рисунке 2.3.

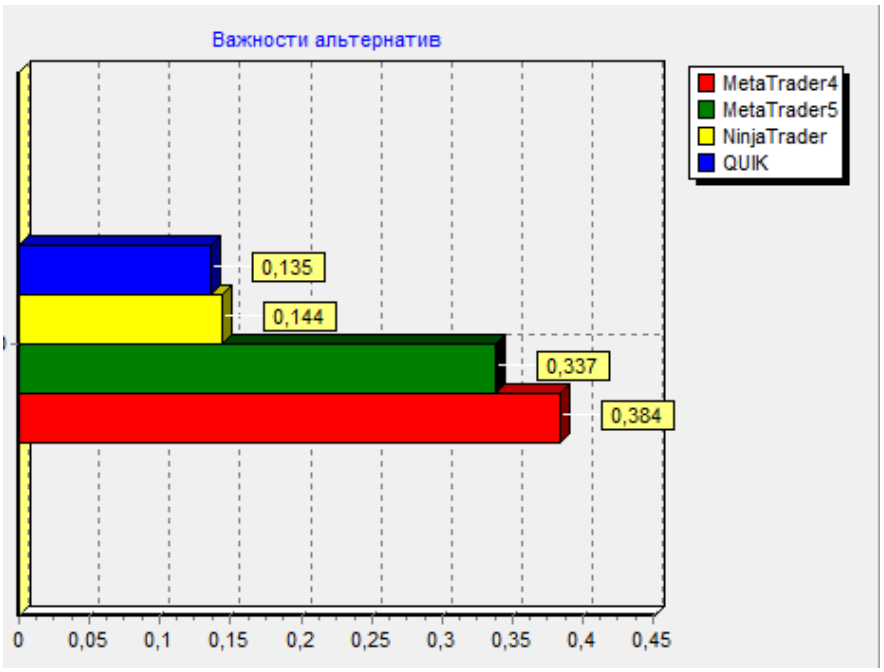


Рисунок 2.3 – Синтез результатов

Если общая согласованность не превышается значение в 0.1, то решение считается достоверным. На рисунке 2.4 представлены результат общей согласованности и список альтернатив с значением их важности.

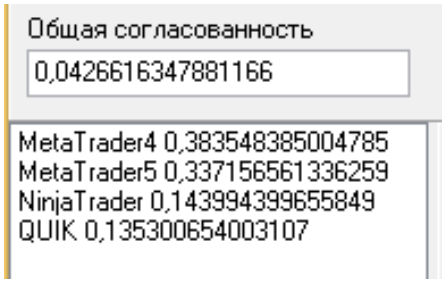


Рисунок 2.4 – Общая согласованность

На рисунке 2.4 видно преимущество MetaTrader 4, но необходимо дать обоснование критериев.

Доступность. NinjaTrader –полный доступ ко всем функциям терминала можно получить только после приобретения премиум аккаунта. Остальные терминалы бесплатно распространяются, но стоит учесть, что QUIK в часто является платной функцией от брокеров.

Популярность. NinjaTrader – очень часто данным терминал используют иностранные брокеры, что сказывается на его популярности.

MetaTrader 5 – данный терминал хоть и предоставляет множество функций и возможностей, но не все брокеры перешли с MT4.

Другие терминалы находятся на другом уровне популярности, что доказывает статистика в 75% брокеров, которые предоставляют доступ к данным терминалам.

Информативность. NinjaTrader – в данном терминале отсутствует справка на русском языке, что усложняет его освоение.

QUIK – справка данного терминала имеет большой объем, но мало информативная. Имея сложный интерфейс отталкивает пользователей, так как необходимо значительное время для освоения.

MetaTrader 4 и 5 – имея понятный интерфейс и информативную справку, притягивают к себе пользователей. [23-25]

Программируемость. Современные терминалы имеют возможность для создания советников, но у каждого своего языка программирования.

QUIK – язык программирования для данного терминала мало известен, тем сам становясь сложным для освоения.

Производительность. Торговые терминалы хорошо оптимизированы для пользователей. Но стоит учесть, что большое количество функций в терминале делает его «громоздким», тем самым нагружая компьютер пользователя, что может негативно сказать на процессе торговли.

После просмотров преимуществ и недостатков торговых терминалов, можно выделить наиболее оптимальный вариант в виде MetaTrader 4. MT4 отлично подходит под необходимые условия.

2.1.2 **Обоснование информационного обеспечения**

Для успеха в процессе разработки торгового советника и понимания протекающих процессов в MetaTrader 4, нужно создать модель базы данных.

Безопасность является одним из главных требований к информационному обеспечению. MetaTrader 4. При помощи советников, индикаторов и скриптов к ним можно получить доступ.

Понятия предметной области, взаимосвязи и ограничения описываются в логической модели. Логическая модель данных – начальный прототип будущей базы данных. Используя термины информационных единиц без привязки к какой-либо СУБД и строится логическая модель. Более того, логическая модель данных необязательно должна быть выражена средствами именно реляционной модели данных. ER–диаграммы (Entity–Relationship, диаграммы сущность–связь) в настоящее время одно из основных средств разработки логической модели.

Для выделения основных процессов на финансовом рынке, была построена логическая диаграмма «сущность–связь», представленная на рисунке 2.5.

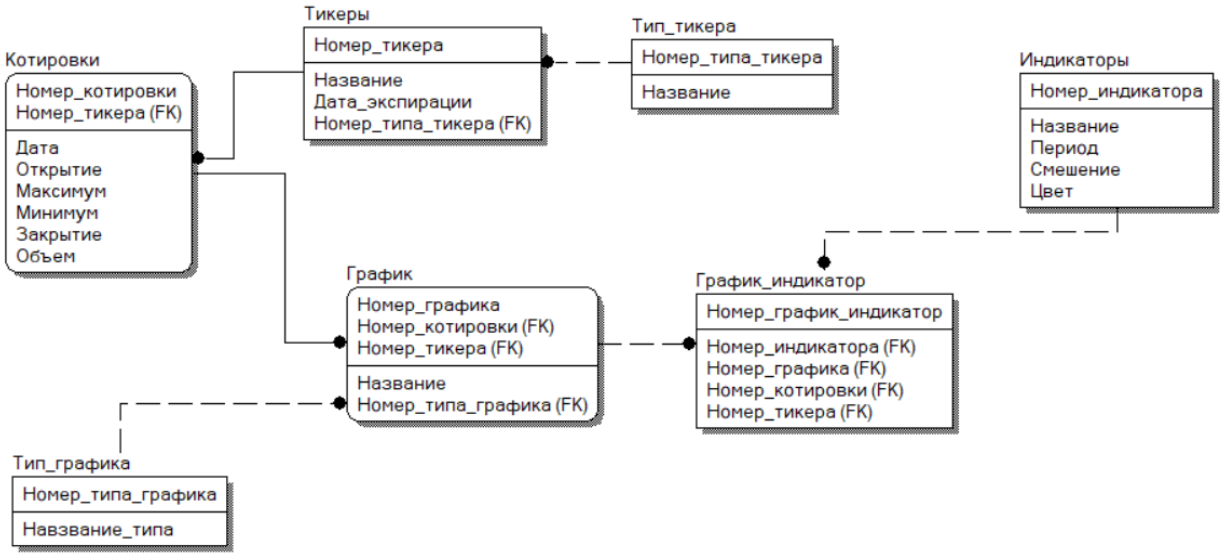


Рисунок 2.5 – Логическая модель базы данных

ER–диаграмма отражает наиболее важные информационные объекты для выбранной предметной области, которые на диаграмме называются сущностями, их свойства – атрибуты и связи между данными объектами [26-29].

На рисунке 2.5 показаны основные сущности:

- тикеры – это короткое название в биржевой информации;
- котировки – это цена единицы торгового инструмента;
- график – отображает котировки пользователю;
- индикаторы – это инструмент для анализа котировок.

2.1.3 **Обоснование технического обеспечения**

Техническое обеспечение представляет собой набор технических средств, компьютерной техники, средств передачи информации, используемых в автоматизированных системах.

После изучения минимальных требований, были выявлены следующие технические устройства для обеспечения оптимальной производимости, предоставленные в таблице 2.1.

Таблица 2.1 – Технические требования

Системный блок	
Материнская плата	ASUS TUF B360
Процессор	Intel Core i5 –5400
Жесткий диск	SATA III, 500 ГБ, 7200rpm
Оперативная память	12 Гб DDR4
Сеть	10/100Мбит,
Видеокарта	Intel Grap 615
Блок питания	500 ВТ
Устройства ввода	
Клавиатура	iMICE 120V
Мышь оптическая	iMICE DT100
Монитор	
Название	Dell E2016H
Диагональ	19.5”
Разрешение экрана	1600x900
Тип матрицы	TN+LED
Интерфейс	VGA

В таблице 2.1 описаны минимальные технические требования к компьютеру. Любая из характеристик может быть улучшена, для более удобного и быстрого использования торгового терминала.

2.2 Информационная модель

Одной из задач выпускной квалификационной работы является разработка торгового советника. Это позволит увеличить прибыль от торговой стратегии, за счет более быстрого, точного и гибкого использования возможностей торгового терминала. После демонстрации модели «Как есть» были выявлены некоторые недостатки этой модели. После чего была построена модель «Как должно быть» которая устраняла выявленные изъяны предыдущей версии.

Для моделирования диаграмм используется методология IDEF0. На диаграммах необходимо отобразить изменения после прошлого анализа в первой главе. Контекстная диаграмма показана на рисунке 2.6.

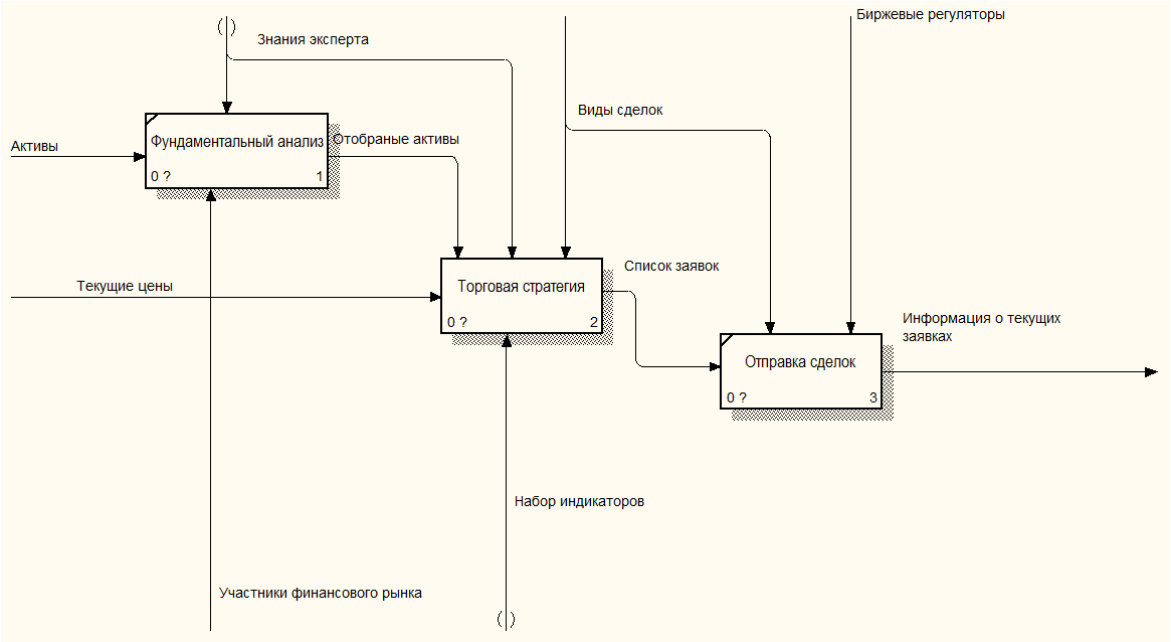


Рисунок 2.6 – Декомпозиция процесса «Отправить заявку»

На рисунке 2.6 можно увидеть, что 2 процесса «Технический анализ» и «Принятие решений» исчезли. Видно, что потоки данных почти не пострадали, но исчезнувшие процессы были автоматизированы. Эти расчеты

будут происходить в торговой стратеги, пользователю больше не нужно выполнять все расчеты вручную.

Далее необходимо отобразить декомпозицию процесса «Торговая стратегия», данная декомпозиция показана на рисунке 2.7.

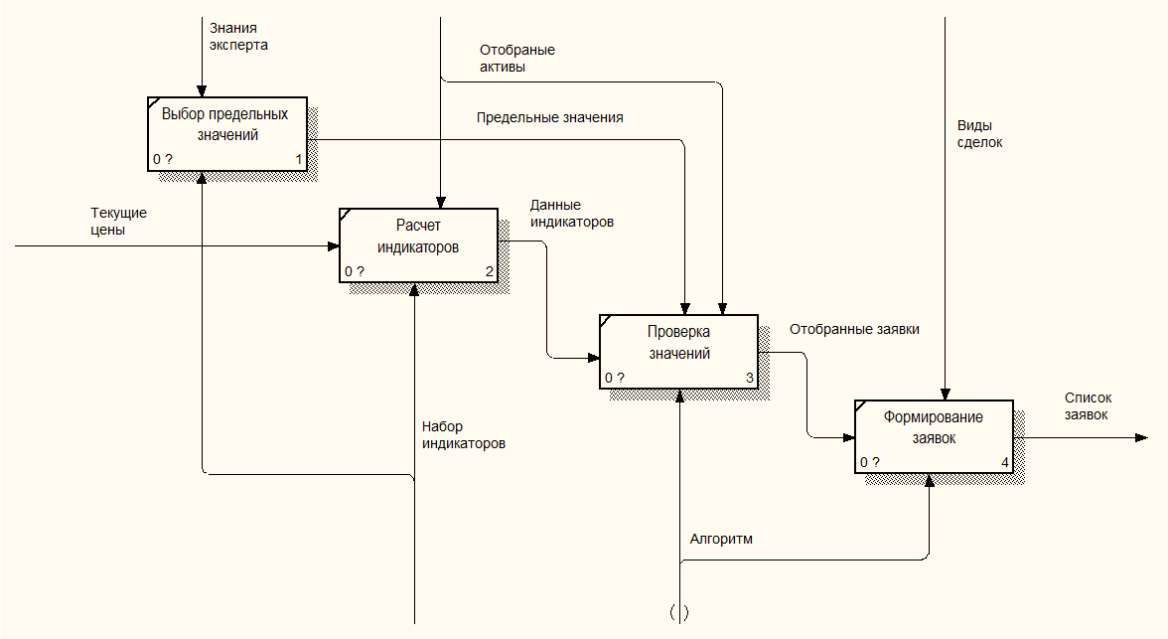


Рисунок 2.7 – Декомпозиция процесса «Торговая стратегия»

На рисунке 2.7 показана декомпозиция, которая отражает основной принцип работы торговой стратегии. После фундаментального анализа пользователем, в данный процесс попадают, только те активы, которые были необходимы пользователю.

Во втором разделе выпускной квалификационной работы, были обоснованы проектные решения по техническому, программному и информационному обеспечению. Был четко определен торговый терминал для осуществления торговли. Также были определенны минимальные технические требования к оборудованию. Построена логическая модель базы данных. Смоделирована диаграмма «Как должно быть».

3 **Программная реализация**

3.1 Построение модели советника

Следующим этапом после изучения торговых терминалов и моделирования является разработка алгоритма работы торгового советника.

Индикатор AO (Awesome Oscillator) – этот индикатор помогает определить текущее поведение движущей силы рынка вычитая значения медленной скользящей среднего (обычно 34 периода) из типичной цены и быстрой скользящей среднего (обычно 5 периодов) из типичной цены. Типичная цена (Typical Price) – частное от деления суммы максимальной цены, минимальной цены и цены закрытия на три.

Метод расчета индикатора AO очень прост – достаточно рассчитать разницу между экспоненциальным средним (EMA) и типичной ценой (TP).

$$\mathrm{TP} = \frac{\mathrm{HIGH} + \mathrm{LOW} + \mathrm{CLOSE}}{3}, \tag{3.1}$$

- high – максимальная цена;
- low – минимальная цена;
- close – цена закрытия.

$$\mathrm{AO} = \mathrm{EMA}\,(\mathrm{TP}, \mathrm{x}) - \mathrm{EMA}\,(\mathrm{TP}, \mathrm{y}), \tag{3.2}$$

- x и y – периоды индикаторов.

Следующий фундаментальный индикатор – линии Фибоначчи, которые помогут легко найти уровни поддержки и сопротивления.

Линии Фибоначчи (LF) строятся следующим образом: во-первых, линия тренда проводится между двумя крайними точками – например, снизу до противоположного пика. После сильного подъема или спада цены часто возвращаются, корректируя значительную часть (а иногда и полностью)

своего первоначального движения. Во время такого обратного движения цены часто встречают поддержку / сопротивление на уровнях линии Фибоначчи или около них.

$$LF = (\text{High} - \text{Low}) * \text{percent} + \text{Low}, \quad (3.3)$$

- high – максимальная цена;
- low – минимальная цена;
- percent – процент Фибоначчи.

После изучения данной информации можно разработать алгоритм для работы торгового советника, который показан на рисунке 3.1.

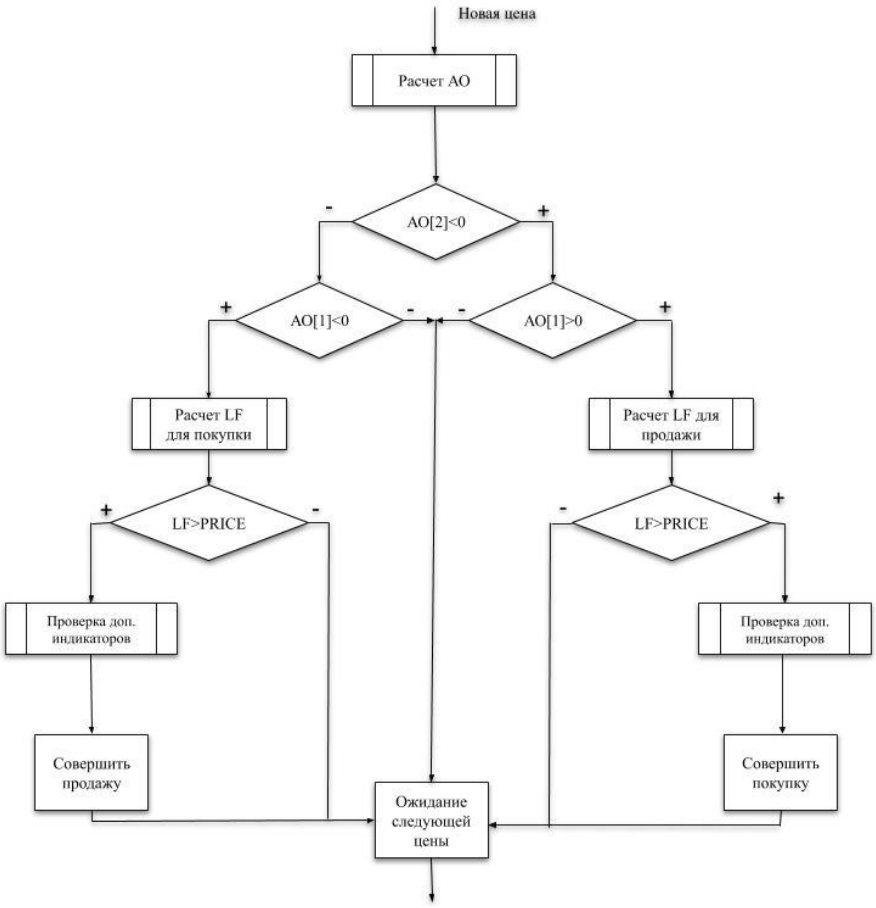


Рисунок 3.1 – Алгоритм торгового советника

На рисунке 3.1 показан алгоритм работы торгового советника, который показывает поведение при получении новой цены. Можно заметить что проверяется вторая свеча (элемент графика для отображения биржевых

котировок за определенный период времени). Если АО преодолел значение 0 в одной из сторон (отрицательное или положительное), то проверяется, изменилось ли это движение на более поздней свече, если оно изменилось, то начинаются вычисления линий Фибоначчи, после чего выполняется тест на пробитие цены. В случае, если цена была пробита, открывается сделка купли–продажи.

3.2 Разработка основных индикаторов советника

После построение модели следует приступить за разработку индикаторов для торгового советника внутри торгового терминала MetaTrader. Сначала необходимо создать нового эксперта, для этого нужно открыть MetaEditor, который встроен в терминал. Его можно открыть с помощью кнопки клавиатуры F4 или вызвать нажатием кнопку в верхнем меню терминала, который показан на рисунке 3.2.

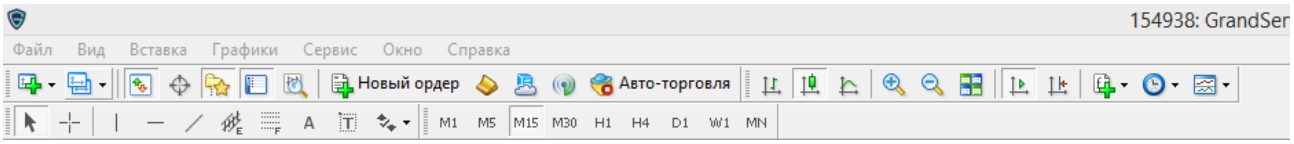


Рисунок 3.2 – Меню MT4

Окно открывшегося MetaEditor, показано на рисунке 3.3.

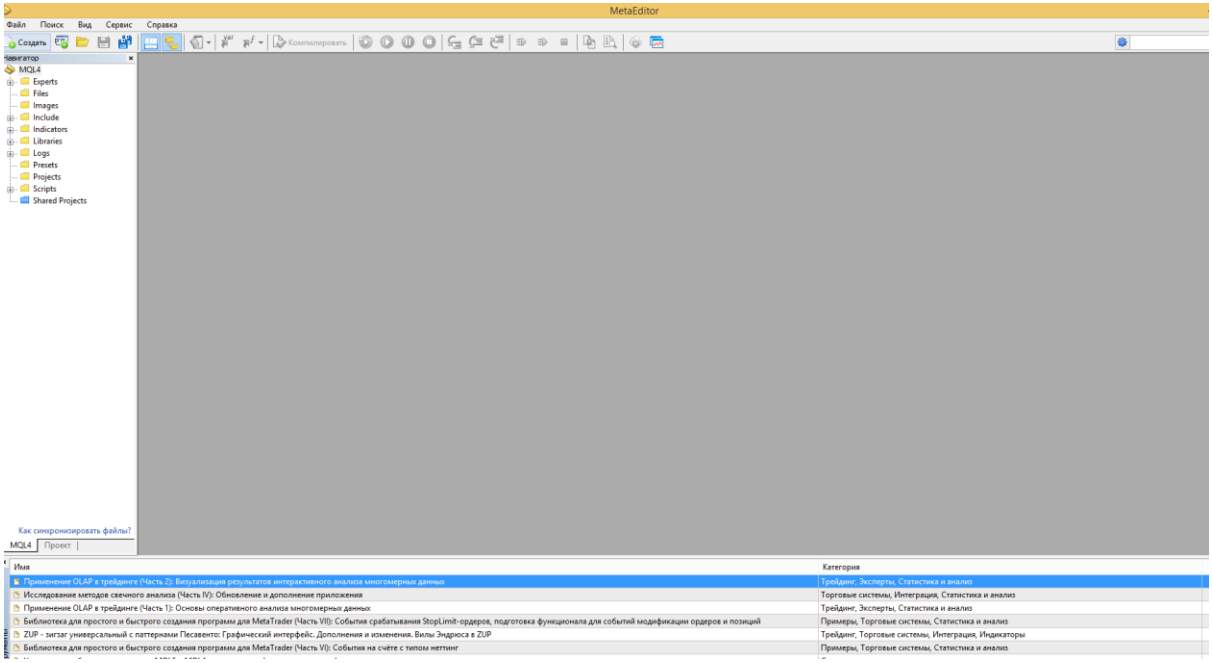


Рисунок 3.3– Окно MetaEditor

На рисунке 3.3 показано окно для редактирования программного кода. Далее нужно создать торгового советника, для этого необходимо нажать на кнопку «Создать», после чего выбрать пункт «Советник (шаблон)».

После выбора советника, необходимо заполнить данные, окно запроса данных показано на рисунке 3.4.

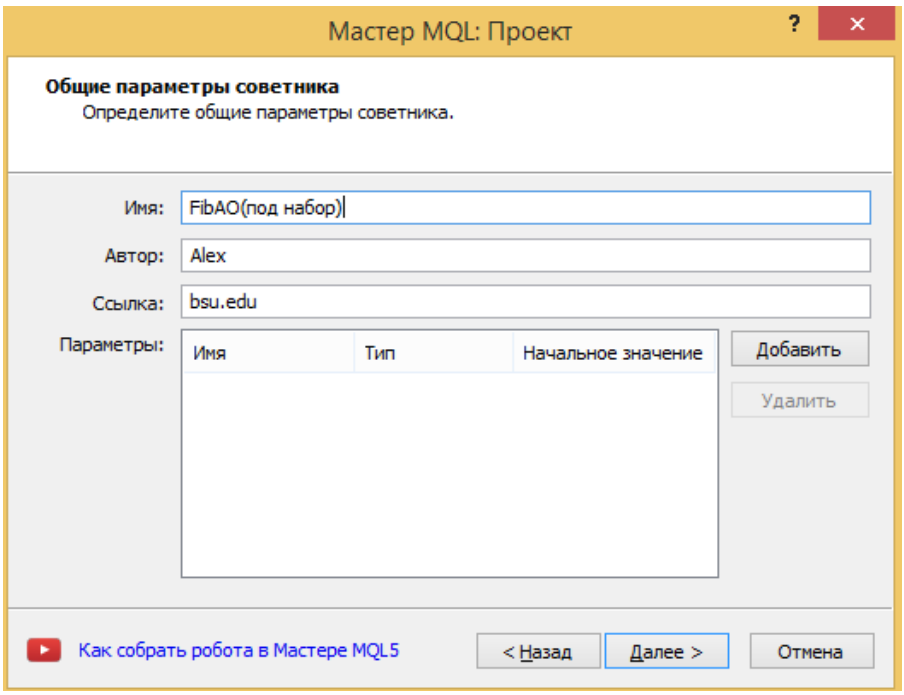


Рисунок 3.4 – Окно для заполнения данных

На рисунке 3.4 окно в котором необходимо заполнить данные о торговом советнике. На рисунке 3.5 показано окно для написания кода.

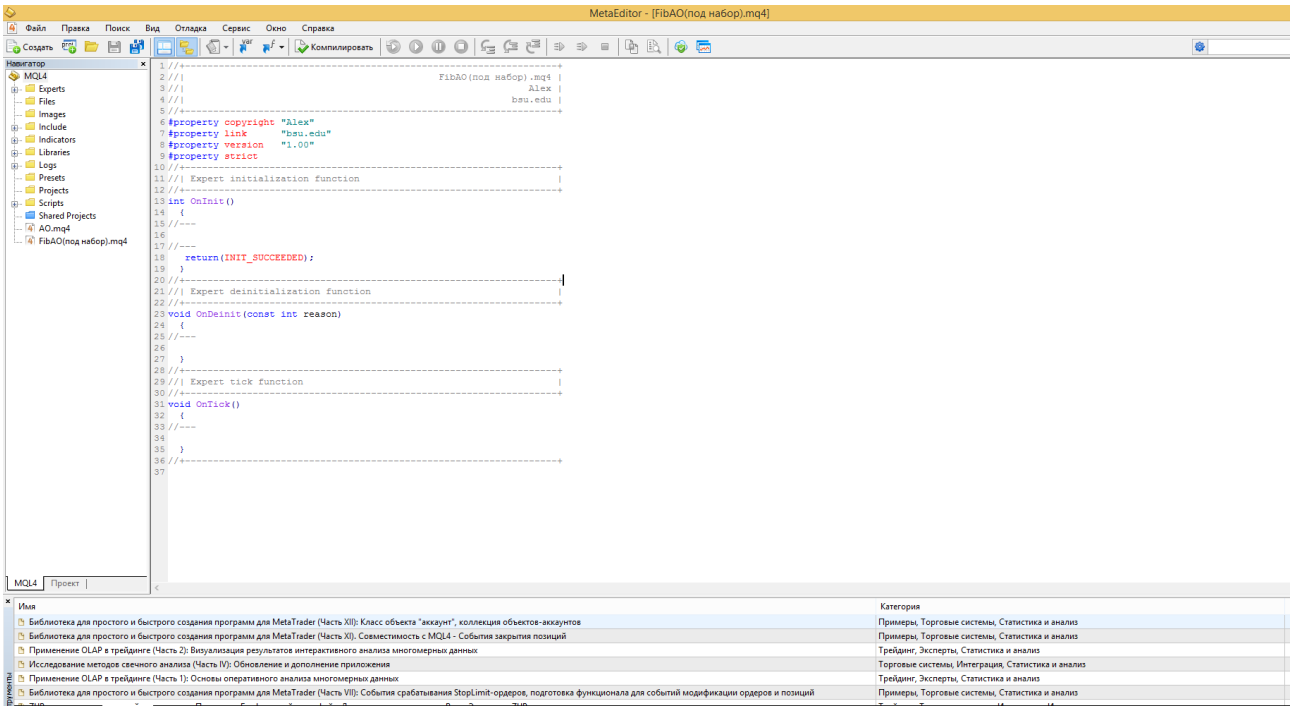


Рисунок 3.5 – Окно редактора для написания программного кода

На рисунке 3.5 изображено окно редактирования, в котором показаны основные функции. Для дальнейшего написания советника, необходимо рассмотреть шаблон.

Необходимо проанализировать следующие функции:

- OnInit – данная функция вызывается при первом запуске торгового советника на торговом счёте;

- OnDeinit – функция, которая вызывается при остановке торговли с помощью советника;

- OnTick – каждая новая свеча вводится в эту функцию MetaTrader.

После создания шаблона торгового советника можно приступать к программированию индикаторов.

Функция расчета линий Фибоначчи отображена на рисунке 3.6.

```
void CalculationFibonacci(int ticket, double up_level, double down_level)
{
   if(ticket != -1){
      DrowFibo(up_level,down_level);
      int array_size = ArraySize(orders_info);
      ArrayResize(orders_info, array_size + 1);
      orders_info[array_size].fibonacci_level[0].level = 1;
      orders_info[array_size].fibonacci_level[1].level = 0;
      orders_info[array_size].fibonacci_level[2].level = 0.236;
      orders_info[array_size].fibonacci_level[3].level = 0.382;
      orders_info[array_size].fibonacci_level[4].level = 0.50;
      orders_info[array_size].fibonacci_level[5].level = 0.618;
      orders_info[array_size].fibonacci_level[6].level = 1.618;
      orders_info[array_size].fibonacci_level[7].level = 2.618;
      orders_info[array_size].fibonacci_level[8].level = 4.236;
      orders_info[array_size].level_number = 6;
      orders_info[array_size].ticket = ticket;
      orders_info[array_size].fibonacci_level[0].price = up_level;
      orders_info[array_size].fibonacci_level[1].price = down_level;
      for(int i = 2; i < 9; i++){
         orders_info[array_size].fibonacci_level[i].price = (up_level - down_level)*orders_info[array_size].fibonacci_level[i].level + down_level;
         orders_info[array_size].fibonacci_level[i].price = NormalizeDouble(orders_info[array_size].fibonacci_level[i].price, Digits);
      }
   }
}
```

Рисунок 3.6 – Листинг алгоритма Фибонначи

На рисунке 3.6 представлена реализация алгоритма. Данная функция принимает данные в виде номера сделки, а также High и Low цену, после чего в цикле рассчитываются все уровни и записываются в структуру.

3.3 Разработка алгоритмов работы набора индикаторов

После реализации основных индикаторов торгового советника необходимо разработать основной набор индикаторов. Для начала реализации, необходимо определить необходимые индикаторы, провести их анализ и после чего составить алгоритмы расчета.

В данный набор входят следующие индикаторы:

- Simple Moving Average (SMA).

Технический индикатор Простое Скользящее Среднее (Simple Moving Average, SMA). Скользящая средняя рассчитывается методом вычисления средней цены от общего значения цен по рассматриваемому активу за прошедшие периоды времени. Зачастую SMA анализирует цены закрытия по активу согласно выставленного таймфрейма. Кстати, в настройках данного индикатора всегда можно сменить это значение [30-32].

Simple Moving Average вовсе не предназначен обеспечивать вхождение в рынок строго в его низшей точке, а выход – строго на вершине. Он позволяет

использовать полученные данные в соответствии с текущей тенденцией: покупать после достижения ценами уровня основания и продавать после появления вершины.

Формула расчета SMA отображена в формуле (3.4).

$$SMA = \frac{\text{SUM(CLOSE (i),P)}}{\text{P}}, \tag{3.4}$$

- SUM – сумма;
- CLOSE (i) – цена закрытия текущего периода;
- P – число периодов расчета.

На рисунке 3.7 изображен алгоритм данного индикатора, который отображает основные действия.

Рисунок 3.7 – Алгоритм работы индикатора SMA

На рисунке 3.7 показан изначальный ввод пользователем параметра, после чего создается переменная для хранения среднего значения. После каждой свечи рассчитывается сумма и делится на период, введенный пользователем, и возвращается в торгового советника.

- Bollinger Bands;

Полосы Боллинджера (BB) являются техническим индикатором, который может применяться и использоваться на всех финансовых рынках, включая Forex и CFD. Индикатор отображает две полосы вокруг краткосрочной, скользящей средней (SMA) движения цены (в середине полосы). Особенности данного индикатора заключаются в сдедующем:

- внезапные изменения цен, обычно происходящие после сужения полосы, соответствующего снижению волатильности;

- если цены выходят за границы полосы, следует ожидать продолжения текущей тенденции;

- если за пиками и впадинами за границами полосы следуют пики и впадины внутри полосы, возможен разворот тенденции;

- движение цен, начавшееся от одной из границ полосы, обычно достигает противоположной границы.

Полосы Боллинджера формируются из трех линий. Средняя линия (MIDDLE LINE, ML) – это обычное скользящее среднее.

Формула необходимая для расчета MIDDLE LINE (3.5).

$$\mathrm{ML} = \mathrm{SMA}(\mathrm{CLOSE}, P), \tag{3.5}$$

Верхняя линия (TOP LINE, TL) – это та же средняя линия, смещенная вверх на определенное число стандартных отклонений (D).

Формула необходимая для расчета TOP LINE (3.6).

$$\mathrm{TL} = \mathrm{ML} + (\mathrm{D} * \mathrm{StdDev}), \tag{3.6}$$

Нижняя линия (BOTTOM LINE, BL) – это средняя линия, смещенная вниз на то же число стандартных отклонений.

Формула необходимая для расчета BOTTOM LINE (3.7).

$$BL = ML - (D * StdDev), \quad (3.7)$$

- CLOSE – цена закрытия;
- P – количество периодов, используемых для расчета;
- SMA – простая скользящая средняя;
- SQRT – квадратный корень;
- StdDev – стандартное отклонение.

Формула расчета стандартного отклонения StdDev (3.8).

$$StdDev = SQRT(\frac{SUM((CLOSE - SMA(CLOSE,P))^2,P)}{P}), \quad (3.8)$$

Для рассчета границ полос в качестве средней линии используется 20 – периодное простое скользящее и 2 стандартных отклонений.Среднии длиной менее 10 периодов малоэффективны.

Далее необходимо простроить алгоритм расчета индикатора, алгоритм изображен на рисунке 3.8.

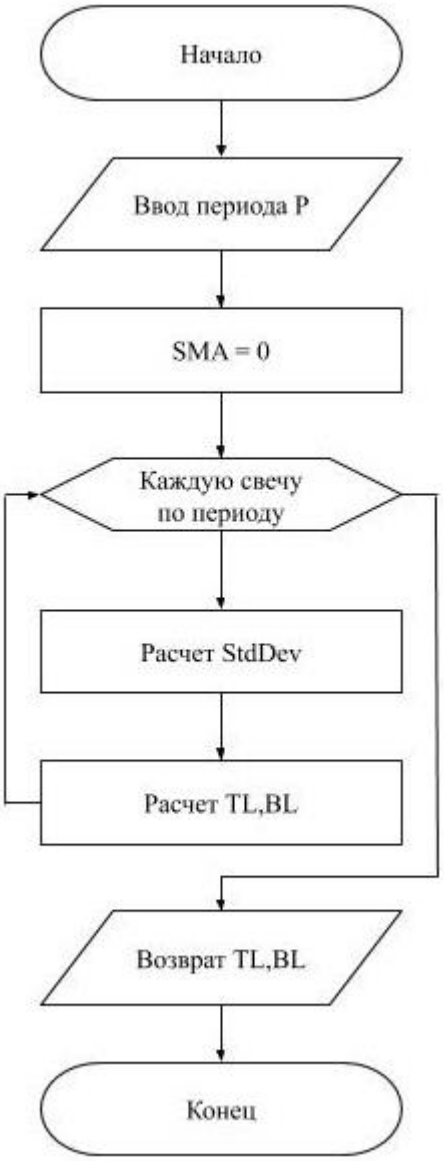


Рисунок 3.8 – Алгоритм расчета индикатора Bollinger Bands

При каждой новой свече индикатор будет рассчитывать за указанный период, после чего возвращать значение в торговую стратегию.

- Relative Strength Index (RSI);

Технический индикатор Индекс Относительной Силы (Relative Strength Index, RSI) – это следующий за ценой осциллятора, который колеблется в диапазоне от 0 до 100. Один из распространенных методов анализа индикатора Relative Strength Index состоит в поиске расхождений, при которых цена образует новый максимум, а RSI не удается преодолеть уровень своего предыдущего максимума [33-35].

Основная формула расчета технического индикатора Relative Strength Index (3.9).

$$\%K = 100 - \left(\frac{100}{1+\frac{U}{D}}\right), \quad (3.9)$$

- U – среднее значение положительных ценовых изменений;
- D – среднее значение отрицательных ценовых изменений.

Алгоритм работы индикатора RSI изображен на рисунке 3.9.

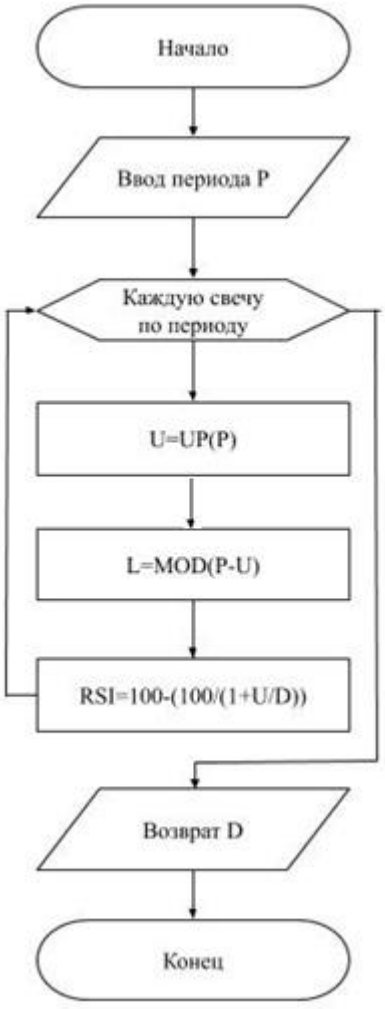


Рисунок 3.9 – Алгоритм работы индикатора RSI

На рисунке 3.9 видно подобно индикатору Bollinger Bands вызывается функция расчета положительных, далее считает средние показатели. После чего остается рассчитать только формулу и вывести на экран.

Необходимо заполнить основные данные и дополнительные параметры. После чего создается шаблон программного кода с наличием главных функций в нём. Пример такого листинга изображено на рисунке 3.10.

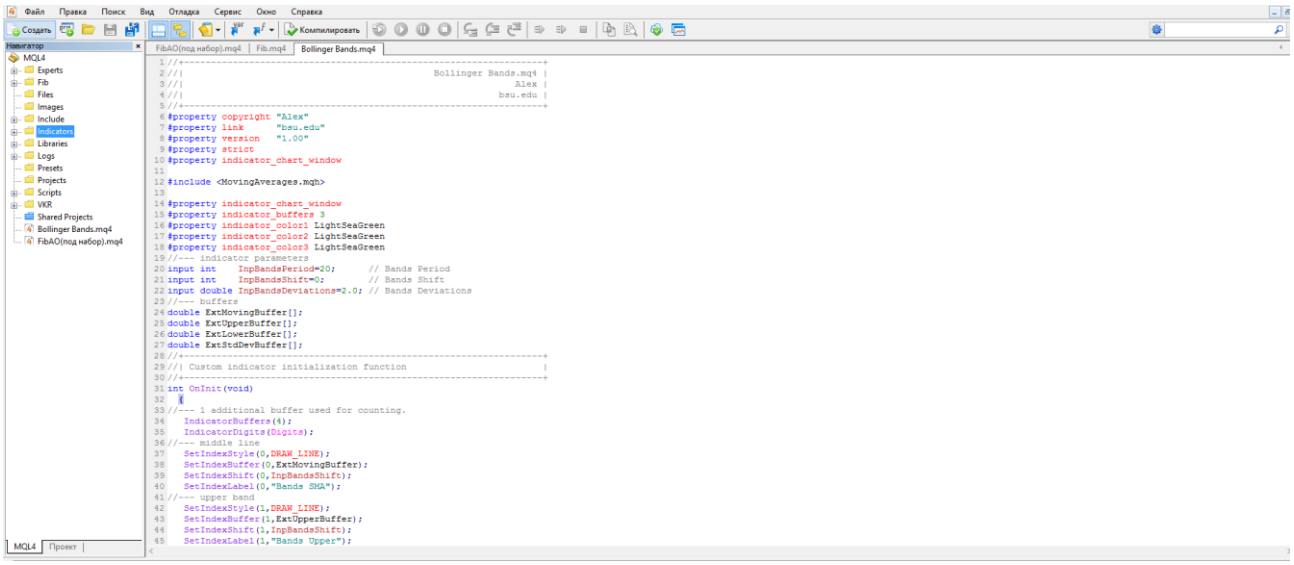


Рисунок 3.10 – Листинг шаблона

На рисунке 3.10 показан шаблон пользовательского индикатора отличающегося от шаблона торгового советника. Была создана новая функция OnCalculate. Данная функция служит для вызова при каждой новой свече. Также можно заметить большое количество параметров, которые отвечают за подсчет рассчитанных свечей, а также данные, хранящиеся на текущей свече.

Теперь необходимо осуществить реализацию данных индикаторов. Листинги данных индикаторов указаны в приложении Б.

3.4 Тестирование

Необходимо провести тестирование разработанного торгового робота. Для проведения тестирования необходимо выполнить вход в торговый

терминал MetaTrader и перейти в режим «Тестер стратегий», изображение данного меню подчеркнуто красной линей на рисунке 3.11.

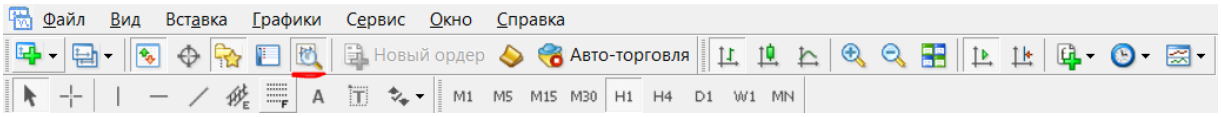


Рисунок 3.11 –Тестировщик стратегий

На рисунке 3.11 показана кнопка для включения тестера стратегий. Окно для тестирования показано на рисунке 3.12.

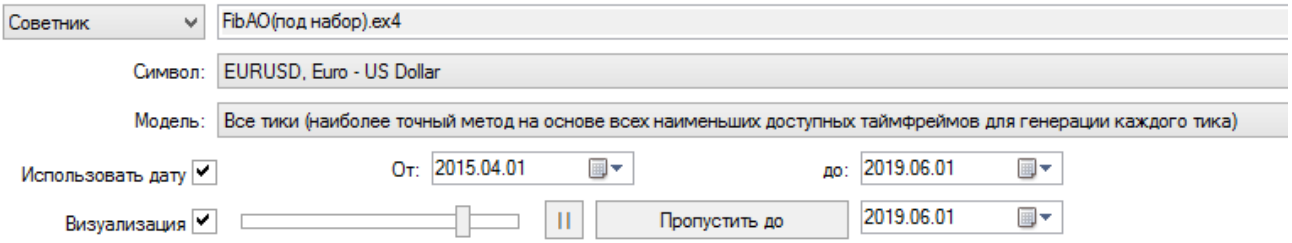


Рисунок 3.12 – Окно тестирования

На рисунке 3.12 показано окно тестирования. В окне тестирования присутствует выпадающий список для выбора объекта тестирования. После необходимо выбрать интересующий объект для тестирования. Для тестирования торгового советника есть ряд настраиваемых параметров [36-38]:

- выбор торгового инструмента в пункте «Символ»;
- выбор способа расчета цены в пункте «Модель»;
- выбор интервала времени для тестирования;
- визуализация для отображения сделок в «реальном времени»;
- определение периода свечей;
- «спред», позволяющий выбрать разницу между ценой покупки и продажи.

На рисунке 3.13 представлены настройки свойств эксперта.

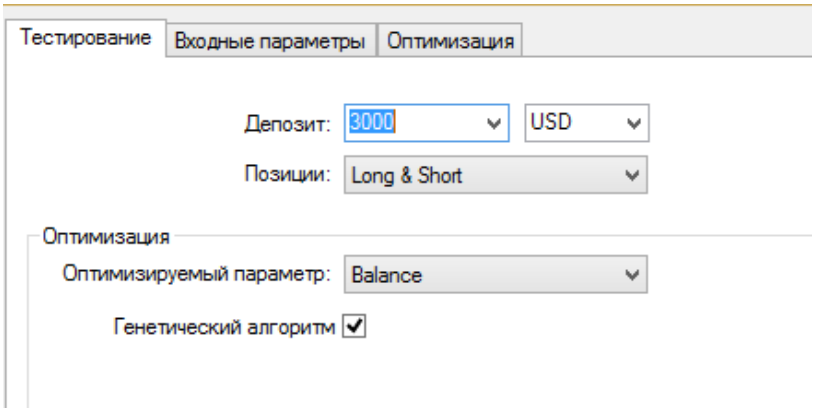


Рисунок 3.13 – Свойства эксперта

На рисунке 3.13 видно три вкладки:

- «тестирование» позволяет указать валюту тестирования, начальный баланс;

- «входные параметры», предоставляют редактирование параметров эксперта;

- «оптимизация» указываются различные пороговые значения.

«Свойства символа» предоставляет информацию о финансовом инструменте [39,40]. В главном окне MetaTrader появляется график с сделками, полученными в результате тестирования при нажатии кнопки.

Для начала тестирования необходимо нажать «Старт». Ниже можно увидеть 4 вкладки, рассмотрим их:

- результаты, отображает все исполненные сделки;

- график, отображает все сделки на графике;

- отчёт, отображает информацию о статистических результатах тестирования;

- журнал, в нём ведется регистрация всех действий.

Далее необходимо провести анализ тестирования. График тестирования показан на рисунке 3.14.

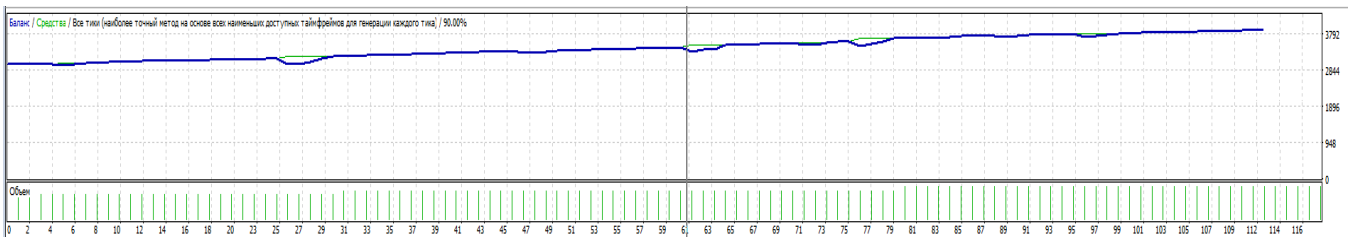


Рисунок 3.14– График тестирования

На графике тестирования хорошо видно рост баланса, несмотря на дальнейшие шаги по увеличению прибыли движение не стабильно.

3.5 Анализ экономической эффективности

После реализации торгового советника необходимо проанализировать экономическую эффективность. Результат работы торгового советника вычисляется соотношением прибыли и затрат на начало деятельности.

Для определения эффективности торгового советника необходимо провести тестирование его на протяжении нескольких пройденных лет. Для просмотра полученных результатов в ходе торговли после добавления индикаторов, необходимо воспользоваться вкладкой «результаты». На рисунке 3.15 показаны результаты торговли.

Объем	Цена	S / L	T / P	Прибыль	Баланс
0.09	1.23629			10.80	3865.03
0.09	1.23358				
0.09	1.23498			10.68	3875.71
0.09	1.23669				
0.09	1.23853				
0.09	1.23703			-2.43	3873.27
0.09	1.23703			13.50	3886.77
0.09	1.23567				
0.09	1.23687			10.80	3897.57

Рисунок 3.15 – Результат торговли

После необходимо сравнить результаты торговых советников для определения наиболее прибыльного.

Сравнение торговых советников по показателям прибыли, количества успешных и убыточных сделок показано в таблице 3.1.

Таблица 3.1 – Сравнение торговых советников

Показатель	FibAO(под набор)	Fireball EA	SZ Scalpel	Yellow
Прибыль	897	-231	143	-2
Количество успешных сделок	73.29%	46.32%	53.2%	49.9%
Сумма для начала	3000	3000	3000	3000
Количество убыточных сделок	26.71%	53.68%	46.8%	50.1%

В таблице 3.1 показано сравнение торговых советников на EURUSD с текущим спредом и периодом в H1. Проанализировав полученные данные можно сделать вывод, что созданный торговый советник приносит большую финансовую прибыль, чем другие.

Таким образом поставленная цель в виде повышения финансовой прибыльности является выполненной.

ЗАКЛЮЧЕНИЕ

В результате выполнения выпускной квалификационной работы цель исследования была достигнута. Финансовая прибыльность торговой стратегии с помощью торгового советника была повышена. Для достижения поставленной цели были выполнены следующие задачи:

- произведен анализ предметной области и выявлены недостатки торговой системы;
- выполнена разработка алгоритмов работы индикаторов;
- осуществлена разработка набора индикаторов;
- выполнено тестирование индикаторов и торговой стратегии с использованием индикаторов;
- проанализирована экономическая эффективность усовершенствованной торговой стратегии.

Разработанный торговый советник доказал эффективность при помощи сравнения его прибыли с другими торговыми советниками. Это доказывает, что данный советник имеет хорошую возможность для использования на финансовых рынках. Также в процессе был разработан набор индикатор, который в дальнейшем может поспособствовать увеличению извлекаемой прибыли, но стоит учесть, что на процесс торговли влияют многие факторы. Если советник показал эффективность в данный момент времени, это не означает, что спустя время он будет приносить прибыль.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Словарь трейдера. Основные понятия [Электронный ресурс]: URL: http://success –everywhere.ru/finansi/birzha–i–rinok/slovar_trejdera (Дата обращения 30.03.2019).
2. Свободная энциклопедия [Электронный ресурс]: URL:https://ru.wikipedia.org/ (Дата обращения 05.04.2019).
3. Московская биржа [Электронный ресурс]: URL:https://www.moex.com (Дата обращения 06.04.2019).
4. Торговый терминал NinjaTrader [Электронный ресурс]: URL:https://ninjatrader.com/ru/ (Дата обращения 10.04.2019).
5. Кондаков К.Г. MetaTrader 4. Учимся зарабатывать на FOREX [Текст] / К.Г. Кондаков, О.В. Бондарь – Москва: Бослен, 2015. – 152с.
6. Торговый терминал MetaTrader 5 [Электронный ресурс]: URL:https://www.metatrader5.com/ru (Дата обращения 11.04.2019).
7. Торговый терминал QUIK [Электронный ресурс]: URL:https://arqatech.com/ru/products/quik/ (Дата обращения 13.04.2019).
8. Построение инфологической модели [Электронный ресурс]: URL:http://citforum.ru/database/dbguide/5–2.shtml (Дата обращения 14.04.2019).
9. Математика в трейдинге. Оценка результатов торговых сделок [Электронный ресурс]: URL: https://www.mql5.com/ru/articles/1492 (Дата обращения 15.04.2019).
10. Голдрат М.Э. Цель –2. Дело не в везении [Текст] / М.Э. Голдрат – Москва: Альбина Паблишер, 2018. – 230с.
11. Как использовать фактор восстановления [Электронный ресурс]: URL:https://freshforex.org/encyclopedia–forex/how_to_use_the_recovery_factor (Дата обращения 16.04.2019).

12. Мезенцев, К.Н. Автоматизированные информационные системы: Учебник для студентов учреждений среднего профессионального образования [Текст] / К.Н. Мезенцев. – Москва: ИЦ Академия, 2015. – 176 с.

13. Мэрфи, Д. Д. Технический анализ финансовых рынков. Полный справочник по методам и практике трейдинга. [Текст] / Д.Д. Мэрфи – Москва: Вильямс, 2016. - 496с.

14. Антамошин, А.Н. Интеллектуальные системы управления организационно-техническими системами [Текст] / А.Н. Антамошин, О.В. Близнова, А.В. Бобов, Большак. - М.: РиС, 2016. - 160 с.

15. Гахов, Р.П. Компьютерное моделирование экономических процессов [Текст]: учебное пособие для студентов вузов / Р.П. Гахов, Н.В. Щербинина. - Белгород: ИД Белгород, 2014. - 88 с.

16. Гахова, Н.Н. Инструментальные средства информационных систем [Электронный ресурс] / Н.Н. Гахова – Белгород: НИУ БелГУ, 2012.: URL:http://pegas.bsu.edu.ru/course/view.php?id=5188 (Дата обращения 19.04.2019).

17. Илюшечкин, В.М. Основы использования и проектирования баз данных [Текст] / В.М. Илюшечкин. – Москва: Юрайт, 2017. – 214 с.

18. Румбешт, В.В. Программирование информационных систем: Учебно-методическое пособие по выполнению лабораторных работ [Текст] / В.В. Румбешт, Г.Г. Банчук. – Белгород: Кооперативное образование, 2016. – 165с.

19. Чистов, Д.В. Проектирование информационных систем. Учебник и практикум [Текст] / Д.В. Чистов, П.В. Мельников. – Москва: Юрайт, 2017. – 216с.

20. Антонова, А.С. Автоматизированные информационные системы, базы и банки данных. Вводный курс: Учебное пособие [Текст] / А.С. Антонова. - М.: Гелиос АРВ, 2014. - 368 с.

21. Алиев, С.А. Быстрая разработка программного обеспечения [Текст] / С.А. Алиев. - М., 2015. - 336 с

22. Баранова, Е. Н. Основы информатики и защиты информации [Текст] / Е.Н. Баранова. - М., 2015. – 192 с.

23. Дэвид Паттерсон. Архитектура компьютера и проектирование компьютерных систем [Текст] / Д. Паттерсон. – М., 2014. – 784 с.

24. Документация MetaTrader 4 [Электронный ресурс]: URL:https://docs.mql4.com/ru (Дата обращения 22.04.2019).

25. Технические индикаторы [Электронный ресурс]: URL:https://www.metatrader4.com/ru/trading_platform/help/analytics/tech_indicators (Дата обращения 03.05.2019).

26. Индикаторы forex [Электронный ресурс]: URL:https://tradexperts.ru/indikatory_forex.htm (Дата обращения 06.05.2019).

27. Уильсам, Л. Секреты торговли на фьючерсном рынке [Текст] / Л. Уильсам – Москва: Альпина Паблишер, 2018. – 234с.

28. Голдрат, М.Э. Я так и знал! Розничная торговля и Теория ограничений [Текст] / М.Э. Голдрат, И.А. Эшколи, Д.Б. Лир: Альпина Паблишер, 2018. – 168с.

29. Кауфман П. Системы и методы биржевой торговли [Текст] / П. Кауфман – Москва: Альбина Паблишер, 2017. – 1279с.

30. Кац, Д. О. Энциклопедия торговых стратегий [Текст] / Д.О. Кац, Д.Л. Маккормик – Москва: Альпина, 2015. – 392с.

31. Шилов Б. MetaTrader: пособие для «кофейников» [Текст] / Б. Шилов, Д. Раннев – Санкт –Петербург: И –Трейд, 2014. – 137с.

32. Лихтенштейн, В.Е. Стандартизация и разработка программных систем: Учебное пособие [Текст] / В.Е. Лихтенштейн. – М: Финансы и статистика, 2014. – 288 с.

33. Лозинский В. С., Егорова Ю. Г. Исследование возможностей алгоритмического трейдинга для повышения эффективности торговых операций [Текст] / Ю.Г. Егорова, 2018. – 211с.

34. Зайцева Е. А. Разработка автоматизированной торговой системы на основе метода системного скальпирования [Текст] / Е.А. Зайцева, 2016. – 157 с.

35. Наздрачева А. Э., Ловянников Д. Г., Галстян А. Ш. Влияние торговых роботов на функционирование фондового рынка [Текст] / А.Э. Наздрачева, 2016. – 325 с.

36. Ананченко И. В. Разработка системы защиты исполняемых файлов ex4 и ex5 терминала metatrader [Текст] / И.В. Ананченко, 2016. – 26 с.

37. Закарян И. Практический интернет-трейдинг. Как работать на рынках акций, фьючерсов, опционов и Forex. – Litres [Текст] / И. Закарян, 2017. – 112 с.

38. Янина О. Н., Локтионова Ю. Н., Давыдова Т. И. Структура и регулирование валютного рынка ФОРЕКС [Текст] / О.Н. Янина, 2019. – 75 с.

39. Хорнер Р. FOREX на 5 часов в неделю. Как зарабатывать трейдингом на финансовом рынке в свое свободное время. – Litres [Текст] / Р. Хорнер, 2017. – 90 с.

40. Лавренова Е. С. Особенности биржевой торговли российского рынка ценных бумаг //Juvenis scientia [Текст] / Е.С. Лавренова, 2016. – 101 с.

ПРИЛОЖЕНИЕ А

Начальные индикаторы

```
#property strict
//--- indicator settings
#property  indicator_separate_window
#property  indicator_buffers 3
#property  indicator_color1  Black
#property  indicator_color2  Green
#property  indicator_color3  Red
//--- buffers
double     ExtAOBuffer[];
double     ExtUpBuffer[];
double     ExtDnBuffer[];
//---
extern int PERIOD_FAST = 5;
extern int PERIOD_SLOW = 34;
//--- bars minimum for calculation
int DATA_LIMIT  = PERIOD_SLOW;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
void OnInit(void)
  {
//--- drawing settings
   SetIndexStyle(0,DRAW_NONE);
   SetIndexStyle(1,DRAW_HISTOGRAM);
   SetIndexStyle(2,DRAW_HISTOGRAM);
   IndicatorDigits(Digits+1);
   SetIndexDrawBegin(0,DATA_LIMIT);
   SetIndexDrawBegin(1,DATA_LIMIT);
   SetIndexDrawBegin(2,DATA_LIMIT);
//--- 3 indicator buffers mapping
   SetIndexBuffer(0,ExtAOBuffer);
   SetIndexBuffer(1,ExtUpBuffer);
   SetIndexBuffer(2,ExtDnBuffer);
//--- name for DataWindow and indicator subwindow label
   IndicatorShortName("AO");
   SetIndexLabel(1,NULL);
   SetIndexLabel(2,NULL);
  }
//+------------------------------------------------------------------+
//| AO                                                               |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &Tteme[],
                const double &Topen[],
                const double &Thigh[],
                const double &lTow[],
                const double &Cclose[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
   int    i,limit=rates_total-prev_calculated;
   double prev=0.0,current;
//--- check for rates total
   if(rates_total<=DATA_LIMIT)
      return(0);
//--- last counted bar will be recounted
   if(prev_calculated>0)
     {
      limit++;
      prev=ExtAOBuffer[limit];
     }
//--- macd
```

```
   for(i=0; i<limit; i++)
     ExtAOBuffer[i]=iMA(NULL,0,PERIOD_FAST,0,MODE_SMA,PRICE_MEDIAN,i)-
                    iMA(NULL,0,PERIOD_SLOW,0,MODE_SMA,PRICE_MEDIAN,i);
//--- dispatch values between 2 buffers
   bool up=true;
   for(i=limit-1; i>=0; i--)
     {
      current=ExtAOBuffer[i];
      if(current>prev)
        up=true;
      if(current<prev)
        up=false;
      if(!up)
        {
         ExtDnBuffer[i]=current;
         ExtUpBuffer[i]=0.0;
        }
      else
        {
         ExtUpBuffer[i]=current;
         ExtDnBuffer[i]=0.0;
        }
      prev=current;
     }
//--- done
   return(rates_total);
  }
//+------------------------------------------------------------------+
```

ПРИЛОЖЕНИЕ Б

Основной набор индикаторов

```
//+------------------------------------------------------------------+
//|                                                     (SMA).mq4 |
//|                                                          Alex |
//|                                                       bsu.edu |
//+------------------------------------------------------------------+
#property copyright "Alex"
#property link      "bsu.edu"
#property version   "1.00"
#property strict
extern string TimeFrameNote="TimeFrame =0 - Current Timeframe, =1 - 1MIN, =5 - 5MIN, =15 - 15MIN, =30 - 30MIN, =60 - 1H, =240 - 4H,
=1440 - D1, =10080 - W1, =43200 - MN1";
extern int TimeFrame=0;
extern int MAPeriod=13;
extern int ma_shift=0;
extern int ma_method=MODE_SMA;
extern int applied_price=PRICE_CLOSE;
//----
double ExtMapBuffer1[];
string Copyright="\xA9 WWW.BEST-METATRADER-INDICATORS.COM";
string MPrefix="FI";
int init()
  {
   string short_name;
//---- indicator line
   SetIndexBuffer(0,ExtMapBuffer1);
   SetIndexStyle(0,DRAW_LINE);
//---- name for DataWindow and indicator subwindow label
   switch(ma_method)
     {
      case 1 : short_name="EMA("; break;
      case 2 : short_name="MTSMMA("; break;
      case 3 : short_name="MTLWMA("; break;
      default : short_name="MTSMA(";
     }
   switch(TimeFrame)
     {
      case 1 : string TimeFrameStr="Period_M1"; break;
      case 5 : TimeFrameStr="Period_M5"; break;
      case 15 : TimeFrameStr="Period_M15"; break;
      case 30 : TimeFrameStr="Period_M30"; break;
      case 60 : TimeFrameStr="Period_H1"; break;
      case 240 : TimeFrameStr="Period_H4"; break;
      case 1440 : TimeFrameStr="Period_D1"; break;
      case 10080 : TimeFrameStr="Period_W1"; break;
      case 43200 : TimeFrameStr="Period_MN1"; break;
      default : TimeFrameStr="Current Timeframe";
     }
   IndicatorShortName(short_name+MAPeriod+") "+TimeFrameStr);
   DL("001", Copyright, 5, 20,Gold,"Arial",10,0);
   return(0);
  }int deinit()
  {   ClearObjects();
   return(0);
  }int start()
  {
   datetime TimeArray[];
   int      i,shift,limit,y=0,counted_bars=IndicatorCounted();     ArrayCopySeries(TimeArray,MODE_TIME,Symbol(),TimeFrame);     limit=Bars-
counted_bars;
   for(i=0,y=0;i<limit;i++)
     {
      if (Time[i]<TimeArray[y]) y++;      ExtMapBuffer1[i]=iMA(NULL,TimeFrame,MAPeriod,ma_shift,ma_method,applied_price,y);
     }   return(0);
  }void DL(string label, string text, int x, int y, color clr, string FontName = "Arial",int FontSize = 12, int typeCorner = 1)
```

```
{
  string labelIndicator = MPrefix + label;
  if (ObjectFind(labelIndicator) == -1)
  {
    ObjectCreate(labelIndicator, OBJ_LABEL, 0, 0, 0);
 }
  ObjectSet(labelIndicator, OBJPROP_CORNER, typeCorner);
  ObjectSet(labelIndicator, OBJPROP_XDISTANCE, x);
  ObjectSet(labelIndicator, OBJPROP_YDISTANCE, y);
  ObjectSetText(labelIndicator, text, FontSize, FontName, clr);
}
void ClearObjects()
{
 for(int i=0;i<ObjectsTotal();i++)
 if(StringFind(ObjectName(i),MPrefix)==0) { ObjectDelete(ObjectName(i)); i--; }
}
//+------------------------------------------------------------------+
//+------------------------------------------------------------------+

//|                                        Bollinger Bands.mq4 |

//|                                                      Alex |

//|                                                   bsu.edu |

//+------------------------------------------------------------------+

#property copyright "Alex"

#property link      "bsu.edu"

#property version   "1.00"

#property strict

#property indicator_chart_window


#include <MovingAverages.mqh>


#property indicator_chart_window

#property indicator_buffers 3

#property indicator_color1 LightSeaGreen

#property indicator_color2 LightSeaGreen

#property indicator_color3 LightSeaGreen

//--- indicator parameters

input int    InpBandsPeriod=20;      // Bands Period

input int    InpBandsShift=0;        // Bands Shift

input double InpBandsDeviations=2.0; // Bands Deviations

//--- buffers

double ExtMovingBuffer[];

double ExtUpperBuffer[];

double ExtLowerBuffer[];

double ExtStdDevBuffer[];

//+------------------------------------------------------------------+

//| Custom indicator initialization function                         |

//+------------------------------------------------------------------+

int OnInit(void)

  {

//--- 1 additional buffer used for counting.
```

```
   IndicatorBuffers(4);
   IndicatorDigits(Digits);
//--- middle line
   SetIndexStyle(0,DRAW_LINE);
   SetIndexBuffer(0,ExtMovingBuffer);
   SetIndexShift(0,InpBandsShift);
   SetIndexLabel(0,"Bands SMA");
//--- upper band
   SetIndexStyle(1,DRAW_LINE);
   SetIndexBuffer(1,ExtUpperBuffer);
   SetIndexShift(1,InpBandsShift);
   SetIndexLabel(1,"Bands Upper");
//--- lower band
   SetIndexStyle(2,DRAW_LINE);
   SetIndexBuffer(2,ExtLowerBuffer);
   SetIndexShift(2,InpBandsShift);
   SetIndexLabel(2,"Bands Lower");
//--- work buffer
   SetIndexBuffer(3,ExtStdDevBuffer);
//--- check for input parameter
   if(InpBandsPeriod<=0)
     {
      Print("Wrong input parameter Bands Period=",InpBandsPeriod);
      return(INIT_FAILED);
     }
//---
   SetIndexDrawBegin(0,InpBandsPeriod+InpBandsShift);
   SetIndexDrawBegin(1,InpBandsPeriod+InpBandsShift);
   SetIndexDrawBegin(2,InpBandsPeriod+InpBandsShift);
//--- initialization done
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
```

```
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//---
   int i,pos;
//---
  if(rates_total<=InpBandsPeriod || InpBandsPeriod<=0)
    return(0);
//--- counting from 0 to rates_total
  ArraySetAsSeries(ExtMovingBuffer,false);
  ArraySetAsSeries(ExtUpperBuffer,false);
  ArraySetAsSeries(ExtLowerBuffer,false);
  ArraySetAsSeries(ExtStdDevBuffer,false);
  ArraySetAsSeries(close,false);
//--- initial zero
  if(prev_calculated<1)
    {
    for(i=0; i<InpBandsPeriod; i++)
      {
      ExtMovingBuffer[i]=EMPTY_VALUE;
      ExtUpperBuffer[i]=EMPTY_VALUE;
      ExtLowerBuffer[i]=EMPTY_VALUE;
      }
    }
//--- starting calculation
  if(prev_calculated>1)
    pos=prev_calculated-1;
  else
    pos=0;
//--- main cycle
  for(i=pos; i<rates_total && !IsStopped(); i++)
    {
    //--- middle line
    ExtMovingBuffer[i]=SimpleMA(i,InpBandsPeriod,close);
    //--- calculate and write down StdDev
    ExtStdDevBuffer[i]=StdDev_Func(i,close,ExtMovingBuffer,InpBandsPeriod);
    //--- upper line
    ExtUpperBuffer[i]=ExtMovingBuffer[i]+InpBandsDeviations*ExtStdDevBuffer[i];
    //--- lower line
    ExtLowerBuffer[i]=ExtMovingBuffer[i]-InpBandsDeviations*ExtStdDevBuffer[i];
    //---
```

```
    }
//---- OnCalculate done. Return new prev_calculated.
  return(rates_total);
 }
//+------------------------------------------------------------------+
//| Calculate Standard Deviation                                     |
//+------------------------------------------------------------------+
double StdDev_Func(int position,const double &price[],const double &MAprice[],int period)
 {
//--- variables
  double StdDev_dTmp=0.0;
//--- check for position
  if(position>=period)
    {
     //--- calcualte StdDev
     for(int i=0; i<period; i++)
        StdDev_dTmp+=MathPow(price[position-i]-MAprice[position],2);
     StdDev_dTmp=MathSqrt(StdDev_dTmp/period);
    }
//--- return calculated value
  return(StdDev_dTmp);
 }
//--- return value of prev_calculated for next call

//+------------------------------------------------------------------+
//|                                                  (SMA).mq4 |
//|                                                      Alex |
//|                                                    bsu.edu |
//+------------------------------------------------------------------+
#property copyright "Alex"
#property link      "bsu.edu"
#property version   "1.00"
#property strict
#property indicator_chart_window
/#property indicator_separate_window
#property indicator_minimum    0
#property indicator_maximum    100
#property indicator_buffers    1
#property indicator_color1     DodgerBlue
#property indicator_level1     30.0
#property indicator_level2     70.0
#property indicator_levelcolor clrSilver
```

```
#property indicator_levelstyle STYLE_DOT
//--- input parameters
input int InpRSIPeriod=14; // RSI Period
//--- buffers
double ExtRSIBuffer[];
double ExtPosBuffer[];
double ExtNegBuffer[];
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit(void)
  {
   string short_name;
//--- 2 additional buffers are used for counting.
   IndicatorBuffers(3);
   SetIndexBuffer(1,ExtPosBuffer);
   SetIndexBuffer(2,ExtNegBuffer);
//--- indicator line
   SetIndexStyle(0,DRAW_LINE);
   SetIndexBuffer(0,ExtRSIBuffer);
//--- name for DataWindow and indicator subwindow label
   short_name="RSI("+string(InpRSIPeriod)+")";
   IndicatorShortName(short_name);
   SetIndexLabel(0,short_name);
//--- check for input
   if(InpRSIPeriod<2)
     {
      Print("Incorrect value for input variable InpRSIPeriod = ",InpRSIPeriod);
      return(INIT_FAILED);
     }
//---
   SetIndexDrawBegin(0,InpRSIPeriod);
//--- initialization done
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Relative Strength Index                                          |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
```

```
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
   int    i,pos;
   double diff;
//---
   if(Bars<=InpRSIPeriod || InpRSIPeriod<2)
     return(0);
//--- counting from 0 to rates_total
   ArraySetAsSeries(ExtRSIBuffer,false);
   ArraySetAsSeries(ExtPosBuffer,false);
   ArraySetAsSeries(ExtNegBuffer,false);
   ArraySetAsSeries(close,false);
//--- preliminary calculations
   pos=prev_calculated-1;
   if(pos<=InpRSIPeriod)
     {
      //--- first RSIPeriod values of the indicator are not calculated
      ExtRSIBuffer[0]=0.0;
      ExtPosBuffer[0]=0.0;
      ExtNegBuffer[0]=0.0;
      double sump=0.0;
      double sumn=0.0;
      for(i=1; i<=InpRSIPeriod; i++)
        {
         ExtRSIBuffer[i]=0.0;
         ExtPosBuffer[i]=0.0;
         ExtNegBuffer[i]=0.0;
         diff=close[i]-close[i-1];
         if(diff>0)
            sump+=diff;
         else
            sumn-=diff;
        }
      //--- calculate first visible value
      ExtPosBuffer[InpRSIPeriod]=sump/InpRSIPeriod;
      ExtNegBuffer[InpRSIPeriod]=sumn/InpRSIPeriod;
      if(ExtNegBuffer[InpRSIPeriod]!=0.0)
```

```
      ExtRSIBuffer[InpRSIPeriod]=100.0-(100.0/(1.0+ExtPosBuffer[InpRSIPeriod]/ExtNegBuffer[InpRSIPeriod]));
    else
     {
      if(ExtPosBuffer[InpRSIPeriod]!=0.0)
        ExtRSIBuffer[InpRSIPeriod]=100.0;
      else
        ExtRSIBuffer[InpRSIPeriod]=50.0;
     }
    //--- prepare the position value for main calculation
    pos=InpRSIPeriod+1;
   }
//--- the main loop of calculations
  for(i=pos; i<rates_total && !IsStopped(); i++)
   {
    diff=close[i]-close[i-1];
    ExtPosBuffer[i]=(ExtPosBuffer[i-1]*(InpRSIPeriod-1)+(diff>0.0?diff:0.0))/InpRSIPeriod;
    ExtNegBuffer[i]=(ExtNegBuffer[i-1]*(InpRSIPeriod-1)+(diff<0.0?-diff:0.0))/InpRSIPeriod;
    if(ExtNegBuffer[i]!=0.0)
      ExtRSIBuffer[i]=100.0-100.0/(1+ExtPosBuffer[i]/ExtNegBuffer[i]);
    else
     {
      if(ExtPosBuffer[i]!=0.0)
        ExtRSIBuffer[i]=100.0;
      else
        ExtRSIBuffer[i]=50.0;
     }
   }
//---
  return(rates_total);
 }
//+------------------------------------------------------------------+
```