МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ВЯТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
ИНСТИТУТ МАТЕМАТИКИ И ИНФОРМАЦИОННЫХ СИСТЕМ
ФАКУЛЬТЕТ АВТОМАТИКИ И ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ
КАФЕДРА ЭЛЕКТРОННЫХ ВЫЧИСЛИТЕЛЬНЫХ МАШИН

Направление 09.03.01 - Информатика и вычислительная техника
(код и наименование направления)
Профиль – Программное и аппаратное обеспечение вычислительной техники

Допускаю к защите
Заведующий кафедрой ЭВМ
_____ / Долженкова М.Л. /
(подпись) *(Ф.И.О.)*

Разработка конструктора Telegram-ботов. Часть 1

Пояснительная записка выпускной квалификационной работы
ТПЖА.09.03.01.514 ПЗ

Разработал: студент гр.ИВТб-4301-04-00 _____ / Бушков Б. Д. / _____

Руководитель: к.т.н., доцент _____ / Долженкова М. Л. / _____

Нормоконтролер: к.т.н., доцент кафедры ЭВМ _____ / Скворцов А. А. / _____
(подпись) *(Ф.И.О)* *(дата)*

Киров 2024

Реферат

Бушков Б. Д. Разработка конструктора Telegram-ботов. Часть 1: ТПЖА.09.03.01.514 ПЗ ВКР / ВятГУ, каф. ЭВМ; рук. Долженкова М.Л. – Киров, 2024. – Гр.ч. 8 л. ф.А1; ПЗ 36 с., 17 рис., 1 табл., 14 форм., 1 источников, 1 прил.

КОНСТРУКТОР, TELEGRAM-БОТ, КЛИЕНТСКАЯ ЧАСТЬ, ВИЗУАЛЬНЫЙ РЕДАКТОР, СЕРВЕРНАЯ ЧАСТЬ, HTTP, GOLAND, POSTGRESQL, TYPESCRIPT, SOLID.JS, JSON, HTML, CSS.

Объект выпускной квалификационной работы - программное средство для упрощения создания и управления ботами в мессенджере Telegram.

Целью данной выпускной квалификационной работы является повышение скорости разработки, настройки и управления ботами, что позволит пользователям без специальных навыков программирования создавать эффективных ботов для различных целей.

Результат работы - конструктор Telegram-ботов, который будет предоставлять набор инструментов и функций для создания и настройки ботов, а также предоставлять возможности для их управления.

Содержание

Введение

В современном мире стали популярными такие приложения для быстрого общения как мессенджеры. Таких приложений достаточно много, но большинство пользователей сети интернет все чаще отдают предпочтение мессенджеру Telegram как наиболее удобному и надежному.

У Telegram имеется удобное API для создания ботов. Бот способен выполнять определенные команды, заданные пользователем через интерфейс Telegram. Данный функционал вполне может удовлетворять потребности компании в предоставлении некоторых услуг в разных сферах. Например, спортивные залы являются одной из таких сфер.

Создание ботов — это трудоемкий процесс, требующий квалифицированных программистов, что довольно затратно для бизнеса.

Для решения данной проблемы существуют конструкторы Telegram-ботов, которые предоставляют функции создания, редактирования и управления ботами. К сожалению, большинство таких конструкторов предоставляют ограниченный функционал при бесплатном использовании, а также имеют закрытые способы хранения данных клиентов. Поэтому было принято решение выполнить анализ и разработать конструктор Telegram-ботов без данных недостатков.

1 Анализ предметной области

На данном этапе работы необходимо рассмотреть функции конструкторов Telegram-ботов, провести обзор существующих на данный момент аналогов, рассмотреть их возможности, выявить их недостатки и обосновать актуальность разработки нового конструктора.

1.1 Telegram-боты и конструкторы

Боты в мессенджере Telegram становятся все более популярными и число их пользователей постоянно растет. Они помогают пользователям выполнять типичные рутинные действия в автоматизированном режиме, значительно упрощая им жизнь. Для владельцев же самих ботов они стали незаменимыми помощниками в работе.

Telegram-боты имеют множество плюсов, таких как:

– круглосуточный доступ;
– моментальный ответ на запрос пользователя;
– удобство использования, интуитивно понятный интерфейс;
– не требуется установка дополнительных программ, общение с ботом ведется через мессенджер.

Telegram-бот используют в коммерческой деятельности для следующих сфер и задач:

– развлечения;
– поиск и обмен файлами;
– предоставление новостей;
– утилиты и инструменты;
– интеграция с другими сервисами;
– осуществление онлайн-платежей.

С популярностью ботов стали появляться все больше различных конструкторов, которые позволяют без наличия специальных знаний и навыков создать своего бота всего в несколько кликов.

Конструктором называется NoCode инструмент, который предназначен для быстрого создания ботов без знания каких-либо языков программирования. Иными словами, весь процесс создания – это нажатие тех или иных кнопок и ввода текста (например, название кнопки, текст сообщения и т.д.).

Первое предназначение – упрощение работы. Ведь не все обладают глубокими знаниями и навыками программирования. Когда боты только появились, их могли разрабатывать только программисты, обладающие соответствующим опытом и навыками.

Помимо того, что конструкторы позволяют расширить аудиторию, способную создавать Telegram-ботов, они экономят время разработчикам. При наличии конструктора нет необходимости разрабатывать каждый раз отдельное приложение для выполнения типовых задач, так как конструктор предоставляет необходимый набор инструментов для быстрого создания бота без необходимости писать код.

Но у конструкторов есть некоторые ограничения, например, при их использовании нельзя выйти за рамки возможностей самого конструктора, а также при выходе нового функционала Telegram API, его реализация в конструкторе происходит с некоторой задержкой. Кроме того, боты, реализованные с помощью NoCode решения обычно менее производительные, чем их аналоги, написанные языке программирования.

1.2 Обзор аналогов

В подпунктах данного раздела рассматриваются существующие аналоги. В качестве рассматриваемых аналогов были выбраны приложения, реализующие функционал создания Telegram ботов с помощью конструктора.

1.2.1 Бот-платформа «ManyBot»

Один из наиболее популярных конструкторов. Работает внутри мессенджера Telegram. Он бесплатный и прост в использовании. Бот предоставляет создание

бота с такими возможностями:

– отправка сообщений;
– создание меню;
– автопостинг из VK, Twitter, YouTube;
– поддержка нескольких языков.

Минусы конструктора состоят в том, что нет администрирования бота за пределами мессенджера, наличие рекламного сообщения в созданном боте, малое количество компонентов и их модификаций, а также отсутствие статистических данных по созданному боту.

1.2.2 Конструктор Telegram ботов «Puzzlebot»

Данный конструктор имеет намного больше возможностей, чем предыдущий сервис: удобный личный кабинет, интуитивный интерфейс, имеет намного больше компонентов, позволяющих реализовывать сложных ботов.

Минусы данного решения в том, что на бесплатном тарифе можно создать лишь одного бота и настроить до 15 команд, а также количество участников бота ограничено 150 пользователями.

1.2.3 Конструктор ботов «Botmother»

Довольно мощный сервис по созданию ботов, который имеет удобный интерфейс для администрирования и создания ботов, предоставляет много различных компонентов, возможность просмотра статистики созданного бота.

В бесплатном тарифе предоставляет лишь создание 10 тестовых ботов с ограниченным функционалом.

1.2.4 Сравнение аналогов

В таблице 1 представлено сравнение вышеперечисленных аналогов.

Таблица 1 – Сравнение аналогов

Критерии \ Аналоги	Manybot	Puzzlebot	Botmother
Удобный доступ для администрирования бота	нет	да	да
Изменение порядка вызовов компонентов	нет	да	да
Нет ограничений на использования компонентов	да	да	нет
Отсутствие рекламы	нет	нет	да

Как видно из таблицы 1 существующие решения имеют ряд недостатков. Также конструкторы больше ориентированы на получение прибыли и ограничивают функционал для бесплатного использования.

Учитывая недостатки рассмотренных аналогов разрабатываемое приложение должно обеспечивать следующий функционал:

- возможность создания ботов с помощью конструктора;
- возможность администрирования бота;
- возможность изменения порядка вызовов компонентов;
- отсутствие ограничений на использование компонентов;
- отсутствие рекламы.

1.3 Актуальность разработки

Telegram боты являются функциональными инструментами для многих пользователей, однако для их разработки зачастую требуются навыки программирования, что усложняет их внедрение в бизнес-процессы компаний. Конструкторы Telegram-ботов по большей части решают данную проблему, предоставляя пользователям удобный интерфейс для создания ботов под конкретные задачи.

Большинство компаний предоставляют ограниченный функционал конструктора, а для расширения их возможности требуют дополнительную плату, что не всегда выгодно для конечного пользователя. Поэтому было принято решение о создании нового конструктора, который исключает вышеперечисленные недостатки, предоставляя пользователям свободную платформу для создания ботов.

Выводы

В данном разделе был проведен анализ предметной области и осуществлен обзор аналогов. Из рассмотренных аналогов были выявлены требуемые функциональные возможности разрабатываемого продукта. Также было выявлено, что многие решения имеют ряд недостатков, таких как ограничения на использование компонентов и показ рекламы. Таким образом, данная тематика и разработка конструктора Telegram-ботов является актуальной.

2 Расширенное техническое задание

В данном разделе представлено техническое задание на разработку конструктора Telegram-ботов.

2.1 Краткая характеристика области применения

Программа предназначена для создания Telegram-ботов, которые будут удовлетворять потребности клиентов в создании их бизнес-решений.

2.2 Основание для разработки

Функциональным назначением программы является предоставление клиентам возможности создания Telegram-ботов при помощи визуального редактора и без глубоких знаний языков программирования.

Программа должна эксплуатироваться на серверах клиента. Для использования конечному пользователю предъявляются требования знания процесса развёртывания серверных приложений.

2.3 Требования к структуре

Конструктор должен состоять из двух частей: серверной и клиентской. Серверная часть должна обеспечивать основной функционал конструктора. Клиентская часть предоставляет собой удобный пользовательский интерфейс.

2.4 Требования к серверной части конструктора

В подпунктах данного раздела описываются требования к серверной части конструктора.

2.4.1 Требования к функциональным характеристикам

Серверная часть конструктора должна предоставлять программный интерфейс, который обеспечивает выполнение следующих функций:

- регистрация пользователей;
- аутентификация пользователей;
- создание ботов;
- вывод списка ботов;
- запуск и остановку ботов;
- добавлять компоненты;
- удалять компоненты;
- редактировать содержимое компонента;
- соединять компоненты;
- обслуживать запросы пользователей от запущенных ботов.

2.4.2 Требования к предоставляемому программному интерфейсу

Программный интерфейс, который предоставляется серверной частью конструктора должен удовлетворять следующим требованиям:

- должен быть прост и интуитивен;
- должен быть надежен и доступен;
- должен включать возможность аутентификации и авторизации;
- должен обрабатывать возможные ошибки при запросе пользователя.

2.4.3 Требования к параметрам технических и программных средств

В состав технических средств должен входить компьютер, включающий в тебя:

- 64-разрядный процессор с тактовой частотой не менее 1.0 ГГц;
- не менее 4 гигабайт оперативной памяти;

– не менее 1 гигабайт свободного дискового пространства;
– сетевую карту.

Также для работы севера требуется предустановленная операционная система на базе ядра Linux: Ubuntu 18.04 или старше, Debian 10 или старше.

2.5 Требования к клиентской части конструктора

В подпунктах данного раздела описываются требования к клиентской части конструктора.

2.5.1 Требования к функциональным характеристикам

Клиентская часть конструктора должна иметь возможность формировать и отправлять данные и запросы для выполнения следующих функций:

– регистрация пользователей;
– аутентификация пользователей;
– создание ботов;
– вывод списка ботов;
– запуск и остановку ботов;
– редактирование ботов.

Для работы с содержимым ботов клиентская часть должна содержать визуальный редактор. Редактор предоставляет пользовательский интерфейс для выполнения следующих функций:

– добавление компонентов;
– удаление компонентов;
– редактирование содержимого компонентов;
– соединение компонентов.

2.5.2 Требования к пользовательскому интерфейсу

Интерфейс конструктора Telegram ботов должен состоять из страниц, содержащих разные визуальные элементы, которые предоставляют пользователям возможность взаимодействовать с конструктором.

Также интерфейс должен обеспечивать наглядное, интуитивно понятное представление.

2.5.3 Требования к клиентскому программному обеспечению

Клиентская часть конструктора Telegram-ботов должна быть доступна для полнофункционального использования с помощью следующих браузеров:

- Edge 88.0 и выше;
- Opera 43.0 и выше;
- Mozilla Firefox 55.0;
- Google Chrome 64.0 и выше.

2.6 Стадии разработки

Разработка должна быть проведена в следующих стадиях:

- разработка технического задания;
- проектирование структуры серверной части конструктора;
- проектирование структуры клиентской части конструктора;
- программная реализация серверной части конструктора;
- программная реализация клиентской части конструктора.

Выводы

В данном разделе было составлено расширенное техническое задание. Определены основные требования к разрабатываемому конструктору Telegram-

ботов, такие как требования к структуре, требования к функциональным характеристикам, требования к интерфейсу и к клиентскому аппаратному и программному обеспечению. Также выделены основные стадии разработки.

3 Разработка структуры приложения

На данном этапе работы необходимо в соответствие с требованиями, поставленными в техническом задании, разработать структуру конструктора и описать основные алгоритмы функционирования.

3.1 Разработка общей структуры конструктора

Общая структура конструктора представлена на рисунке 1.

Конструктор состоит из серверной и клиентской части.

Серверная часть состоит из нескольких сервисов:

– сервис пользователей;
– сервис ботов;
– сервис, обслуживающий ботов.

Сервис пользователей взаимодействует с хранилищами в виде базы данных и кэша, в которых хранятся пользователи конструктора и их токены авторизации. Через него происходит регистрация и аутентификация пользователей.

Сервис ботов предоставляет программный интерфейс для работы с ботами. Данный сервис выполняет следующие функции:

– создание ботов;
– редактирование ботов;
– удаление ботов;
– запуск и остановка ботов.

Данные ботов сохраняются в базе данных. Авторизация пользователя происходит с помощью данных из кэша пользователей.

Сервис, обслуживающий ботов, обрабатывает запросы от запущенных Telegram ботов, также он получает и выполняет команды на запуск и остановку ботов от сервиса ботов, команды при этом посылаются через сервер обмена сообщениями. Данный сервис сохраняет и получает текущее состояние пользователя из кэша.

Пользователь взаимодействует с конструктором через веб-интерфейс, ко-

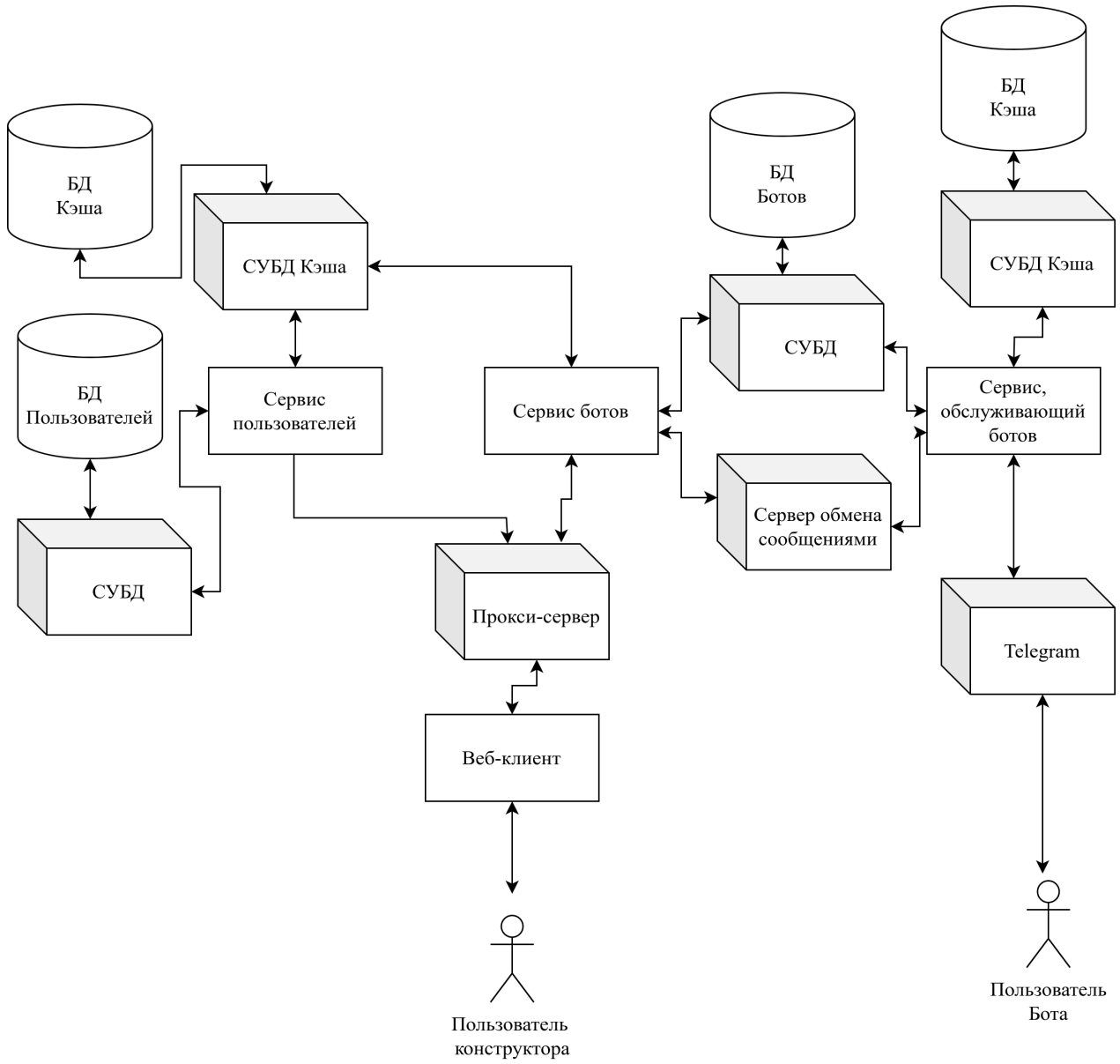

Рисунок 1 – Общая структура конструктора

торый через прокси-сервер связан с сервисами пользователей и ботов. Веб-интерфейс является клиентской частью конструктора.

3.2 Разработка структуры серверной части конструктора

В данном разделе описываются разработанные структуры серверной части конструктора и его основные алгоритмы функционирования.

3.2.1 Модульная структура серверной части конструктора

Модуль – набор структур и методов, который обобщает какую-то логику приложения.

Модульная структура серверной части конструктора представлена на рисунке 2.

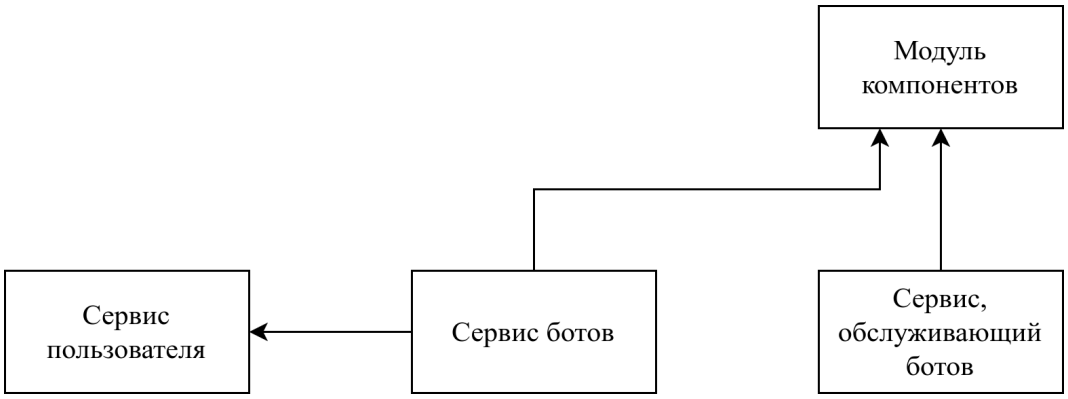

Рисунок 2 – Модульная структура серверной части конструктора

Сервис ботов зависит от сервиса пользователей, который предоставляет первому методы для авторизации пользователя. Также сервис ботов зависит от модуля компонентов, который описывает структуры компонентов и реализует их логику выполнения.

Для выполнения логики ботов сервис, обслуживающий ботов, вызывает методы из модуля компонентов.

3.2.2 Структура модуля компонентов

Модуль компонентов состоит из следующих подмодулей:

- подмодуль компонентов;
- подмодуль контекста;
- подмодуль исполнителя;
- подмодуль ввода-вывода.

Структура модуля компонентов представлена на рисунке 3.

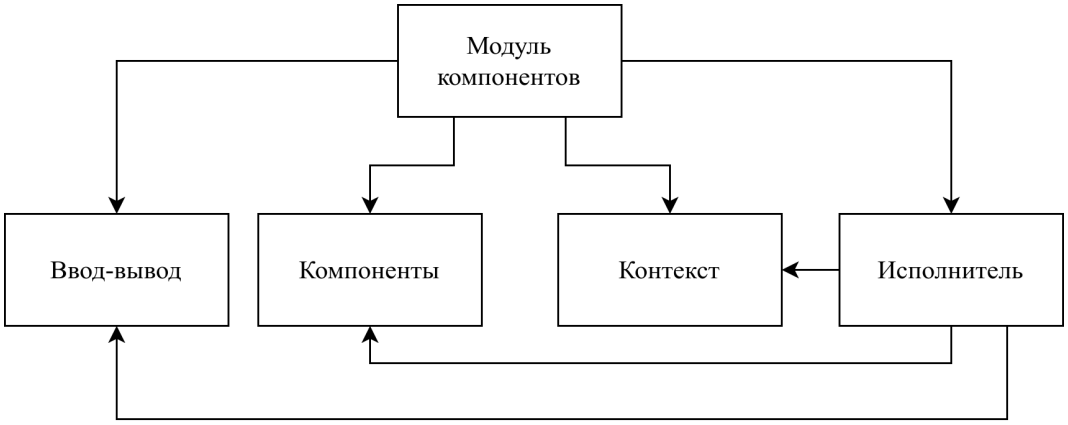

Рисунок 3 – Структура модуля компонентов

Подмодуль компонентов содержит структуру и реализацию логики компонентов. Каждый компонент реализует общий интерфейс компонента, который определен в данном подмодуле.

В данном подмодуле определены следующие компоненты:

– компонент ввода текста;
– компонент отправки сообщения;
– компонент вывода кнопок;
– компонент форматирования;
– компонент точки входа;
– компонент условия.

Подмодуль ввода-вывода предоставляет интерфейс, через который окружение обменивается данными с рядом компонентов.

Подмодуль контекста содержит методы для работы с контекстом. Контекст в рамках бота – память, с которой работают компоненты: компоненты получают из контекста данные для выполнения и записывают в него результат.

Исполнитель представляет собой объект, который содержит в себе контекст и интерфейс ввода-вывода. Через него происходит выполнение компонентов.

3.2.3 Алгоритмы функционирования серверной части конструктора

Пользователь конструктора взаимодействует с сервисом ботом, который контролирует изменение данных и состояние ботов. Схема алгоритма обработки запросов от пользователей сервиса ботов представлен на рисунке 4.

При получении запроса проверяется его корректность. Если запрос не корректен, например, такое обращение не доступно, то выводится соответствующая ошибка. Чтобы обработка прошла успешно, пользователь должен быть авторизован в системе. В случае успешной обработки запроса выдается ответ, иначе - ошибка.

При взаимодействии Telegram пользователя с ботом происходит отправка запросов сервису, обслуживающему ботов, который в дальнейшем обрабатывает данное событие. Схема алгоритма обработки запросов от пользователя бота представлен на рисунке 5.

При принятии события сервисом происходит считывание следующих данных:

– информация о пользователе;
– информация о чате;
– id бота, от которого пришло сообщение.

На основе этих данных из хранилища идёт получение следующих данных:

– контекст пользователя;
– id текущего компонента пользователя;
– компонентов бота.

На основании id текущего компонента происходит получение текущего компонента, который затем выполняется. Результат выполнения представляет собой id следующего компонента, который присваивается текущему.

На основании следующего компонента принимается решение: если компонент ожидает ввода каких-либо данных, то алгоритм заканчивается с сохранением контекста и id текущего компонента, иначе идет выполнение следующего компонента.

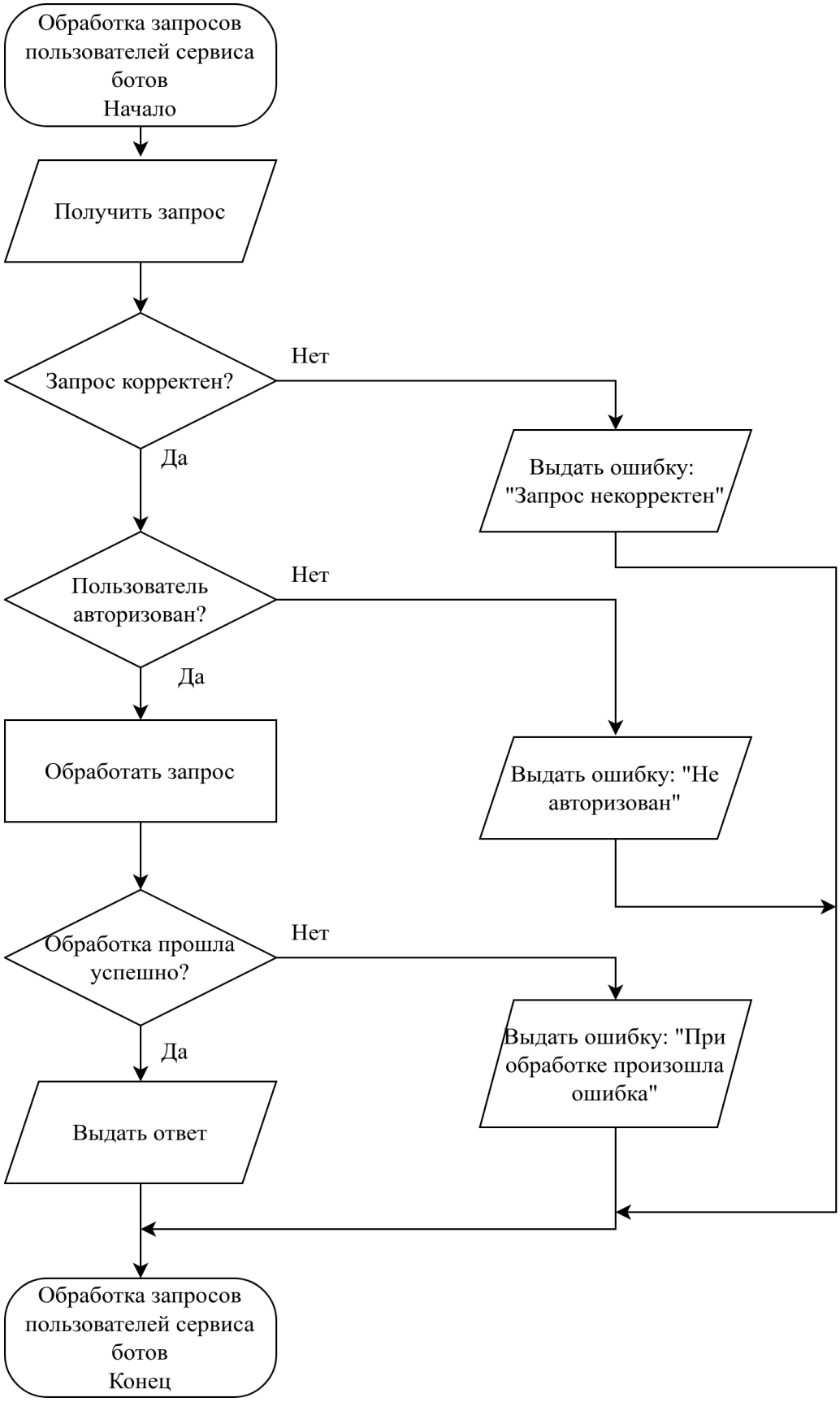

Рисунок 4 – Схема алгоритма обработки запросов от пользователей сервиса ботов

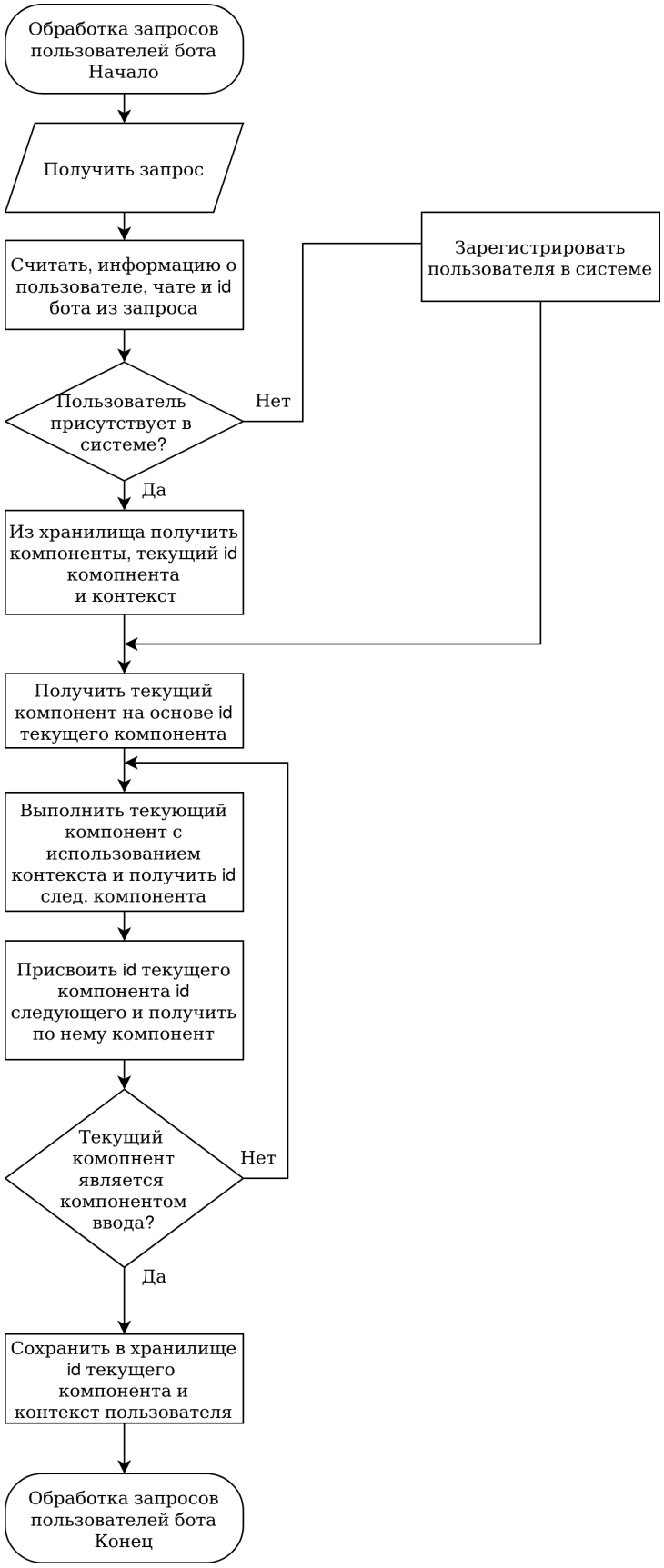

Рисунок 5 – Схема алгоритма обработки запросов от пользователей ботов

3.2.4 Обеспечение защиты информации клиентов конструктора

Для обеспечения защищенного хранения паролей в базе данных используется адаптивная криптографическая хеш-функция bcrypt.

Функция основана на шифре Blowfish. Для защиты от атак с помощью радужных таблиц bcrypt использует соль (salt); кроме того, функция является адаптивной, время её работы легко настраивается и её можно замедлить, чтобы усложнить атаку перебором.

Функция принимает три параметра: стоимость(cost), соль и пароль. Алгоритм хеширования состоит из следующих шагов:

1) инициализация состояния Blowfish с помощью дорогостоящего алгоритма настройки ключей; результат состоит из массива P, состоящего из 18 подключей, и четырех S-боксов;
2) шифрование текста «OrpheanBeholderScryDoubt» 64 раза с помощью стандартного Blowfish в режиме простой замены;
3) результат состоит из конкатенации стоимости, соли и зашифрованного текста «OrpheanBeholderScryDoubt».

Дорогостоящий алгоритм настройки ключей включает в себя следующие шаги:

1) инициализация подключей P и S-боксов шестнадцатеричными цифрами числа π;
2) перестановка P и S на основе пароля и соли: ExpandKey(P, S, password, salt);
3) повторение 2^{cost} раз следующих шагов:
   3.1) перестановка P и S на основе пароля: ExpandKey(P, S, password, 0);
   3.2) перестановка P и S на основе соли: ExpandKey(P, S, salt, 0).

Алгоритм ExpendKey:

1) смешивание пароля с массивом подключей P;
2) разбиение соли на две равные части;
3) инициализация буфера для хранения блоков;

4) циклическое смешивание внутреннего состояния с P, используя половины соли;
5) циклическое смешивание зашифрованного состояния с внутренними S-блоками состояния, используя половины соли.

Аутентификация пользователя происходит по паролю и логину. Отправленный пароль после хеширования сравнивается с хешем из базы данных, и в случае успеха генерируется токен доступа. Этот токен временно сохраняется в базе данных, а его копия выдается пользователю. Используя токен, пользователь может получать доступ к функциям сервиса ботов.

3.3 Разработка структуры клиентской части конструктора

В данном разделе описываются разработанные структуры клиентской части конструктора и его диаграммы состояний.

3.3.1 Структура интерфейса конструктора

Пользовательский интерфейс конструктора представляет собой административную панель с набором следующих страниц:

- страница аутентификации;
- страница регистрации;
- страница ботов пользователя конструктора;
- страница создания нового бота;
- страница запуска бота;
- страница редактирования бота.

Каждая страница состоит из верхней панели и содержимого страницы. Верхняя панель содержит ссылки на страницы аутентификации и регистрации, если пользователь не вошёл в систему, иначе - кнопку “выйти из системы”. Шаблон представлен на рисунке 6.

Страница аутентификации и регистрации содержат поля ввода логина и пароля пользователя, под которым располагается кнопка входа или регистрации.

Рисунок 6 – Общий шаблон страниц

Шаблон содержимого страницы аутентификации представлен на рисунке 7.

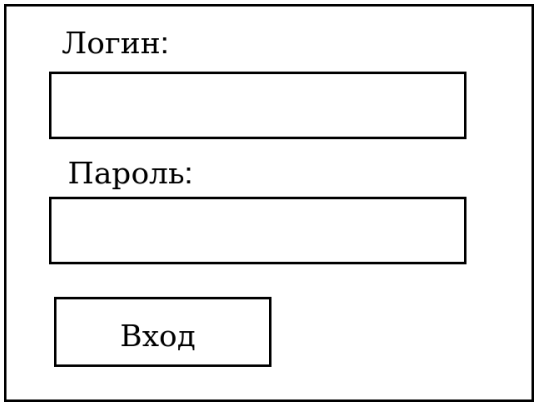

Рисунок 7 – Шаблон содержимого страницы аутентификации

Страница ботов содержит список блочных элементов, которые включают в себя:

- название бота;
- статус бота;
- кнопка для перехода к редактированию бота;
- кнопка запуска или остановки бота.

Шаблон содержимого страницы списка ботов представлен на рисунке 8.

Рисунок 8 – Шаблон содержимого страницы списка ботов

Страница создания бота содержит одно поле ввода, под которым располагается кнопка создания. Шаблон содержимого представлен на рисунке 9.

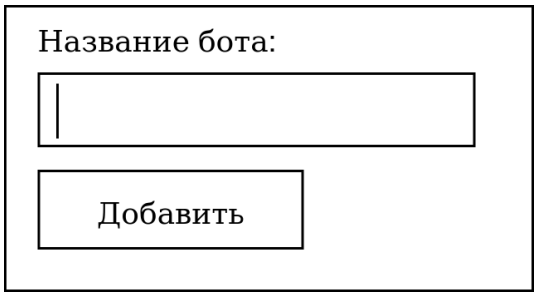

Рисунок 9 – Шаблон содержимого страницы создания бота

Страница запуска бота включает в себя поле ввода токена и кнопку запуска. Шаблон имеет такую же структуру, как и у содержимого страницы создания бота, только с другим именованием кнопки и заголовка поля ввода.

Страница редактирования бота содержит визуальный редактор.

3.3.2 Разработка структуры визуального редактора

Визуальный редактор ботов представляет собой область, на которой пользователь может добавлять, редактировать и удалять компоненты, а также связывать их между собой.

3.3.2.1 Модульная структура редактора

Редактор состоит из следующих модулей (Рисунок 10):

– модуль Api-клиента;
– модуль контроллера;
– модуль представления;
– модуль хранилища.

Рисунок 10 – Модульная структура редактора

API клиент содержит в себе функции обращения к серверу конструктора для получения и обновления данных бота.

Хранилище содержит методы для изменения данных редактора. При изменении данных хранилища происходит обновление их и на сервере через

API клиент. Контроллер служит посредником между представлением и хранилищем: он содержит обработчики, которые меняют состояние редактора. Использует функции API клиента.

Вид редактора (или представление) содержит в себе компоненты редакто-

ра, от которых идут запросы от пользователя. Запросы передаются контроллеру, который их обрабатывает.

3.3.2.2 Компонентная структура редактора

Редактор можно разбить на иерархический набор компонентов: где вышестоящий компонент является родителем, а компонент, который в нем содержится, - дочерним.

Компоненты общаются друг с другом посредством передачи параметров и вызова событий. Родительский компонент вызывает дочерний с помощью передачи параметров, а также отлавливает события дочернего элемента при изменении его состояния.

Компонентная структура визуального редактора представлена на рисунке 11.

На самой высокой ступени стоит компонент редактор. Он хранит все состояние приложения, а также изменяет его с помощью контроллера. Он состоит из области редактора и панели добавления компонентов.

Панель компонентов содержит разные виды компонентов, которые можно добавить на область редактора путем перетаскивания.

В области редактора содержится набор компонентов бота и их соединений.

Компонент бота содержит в себе следующие компоненты:

- содержимое компонента;
- входные точки компонента;
- выходные точки компонента;
- область соединения.

Компоненты включают в себя разные виды содержимого. Содержимое зависит от типа компонента.

Чтобы контролировать переход по компонентам присутствуют элементы соединений – точки соединения. Благодаря им можно располагать линии между компонентами и тем самым связывать их.

Существует три вида точек соединений:

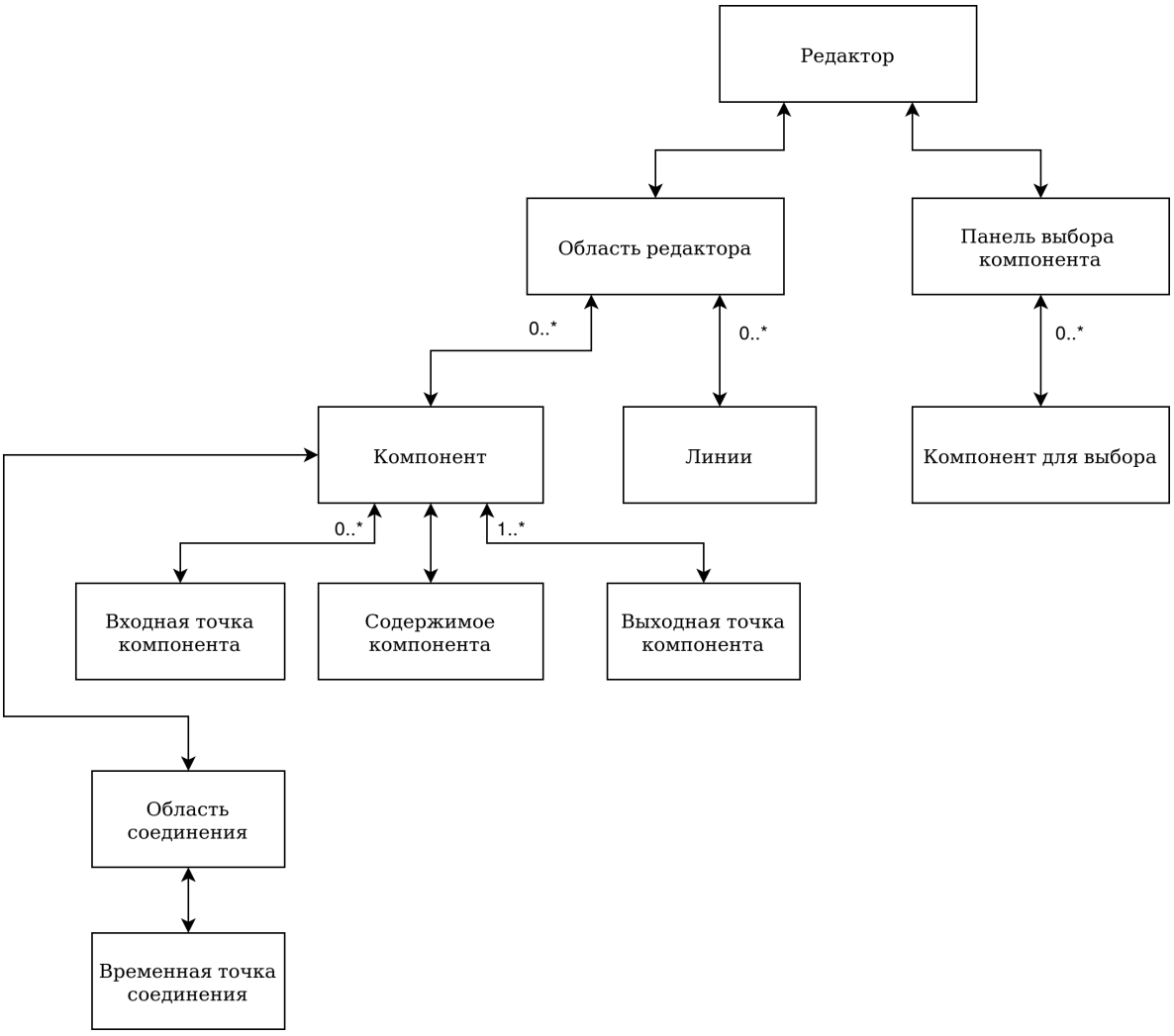

Рисунок 11 – Компонентная структура редактора

– входная точка;
– выходная точка;
– временная точка.

Входная точка. Служит для обозначения места соединения у следующего компонента. Может быть несколько – зависит от количества предыдущих компонентов. При нажатии на данный элемент будет происходить событие отвязки.

Выходная точка. Служит для указания следующего компонента. Может быть несколько - зависит от типа компонента. При нажатии на точку будет происходить событие начала соединения.

Временная точка соединения – элемент, который помогает пользователю

обозначить место соединения у следующего компонента. Данная точка располагается на области соединения компонента. Вызывает событие конца соединения при отжатии левой кнопки мыши на этом элементе. При этом событии происходит скрытие временной и вставка входной точки.

Область соединения компонента представляет собой место, где возможно расположение входных точек. Область охватывает края компонента.

Также у каждого компонента присутствует кнопка удаления.

Расположение компонентов на шаблоне визуального редактора представлено на рисунке 12.

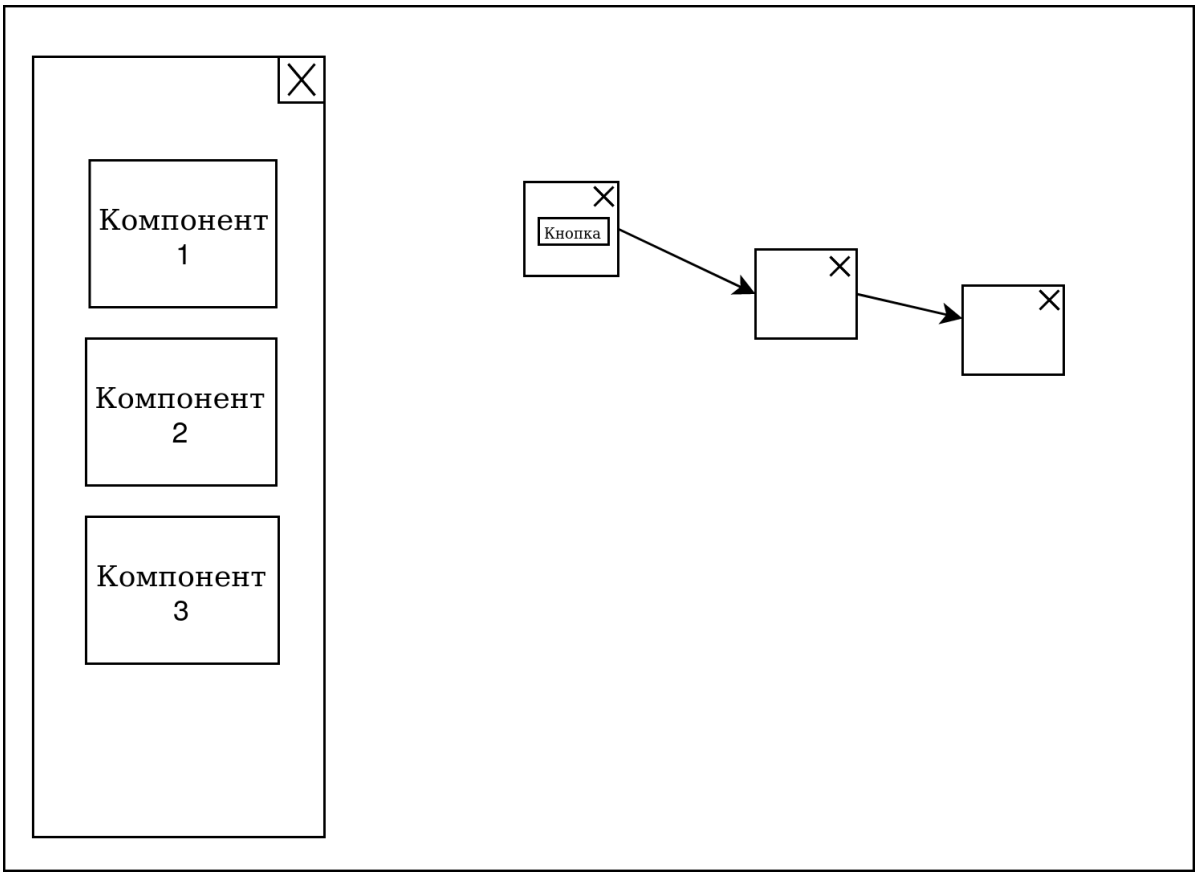

Рисунок 12 – Шаблон визуального редактора

3.3.3 Разработка диаграмм состояний клиентской части конструктора

3.3.3.1 Диаграмма состояний клиентской части

Клиентская часть конструктора имеет определенный набор состояний. Эти состояния представляют собой страницы, на которых может находиться пользователь. Диаграмма состояний клиентской части конструктора представлена на рисунке 13.

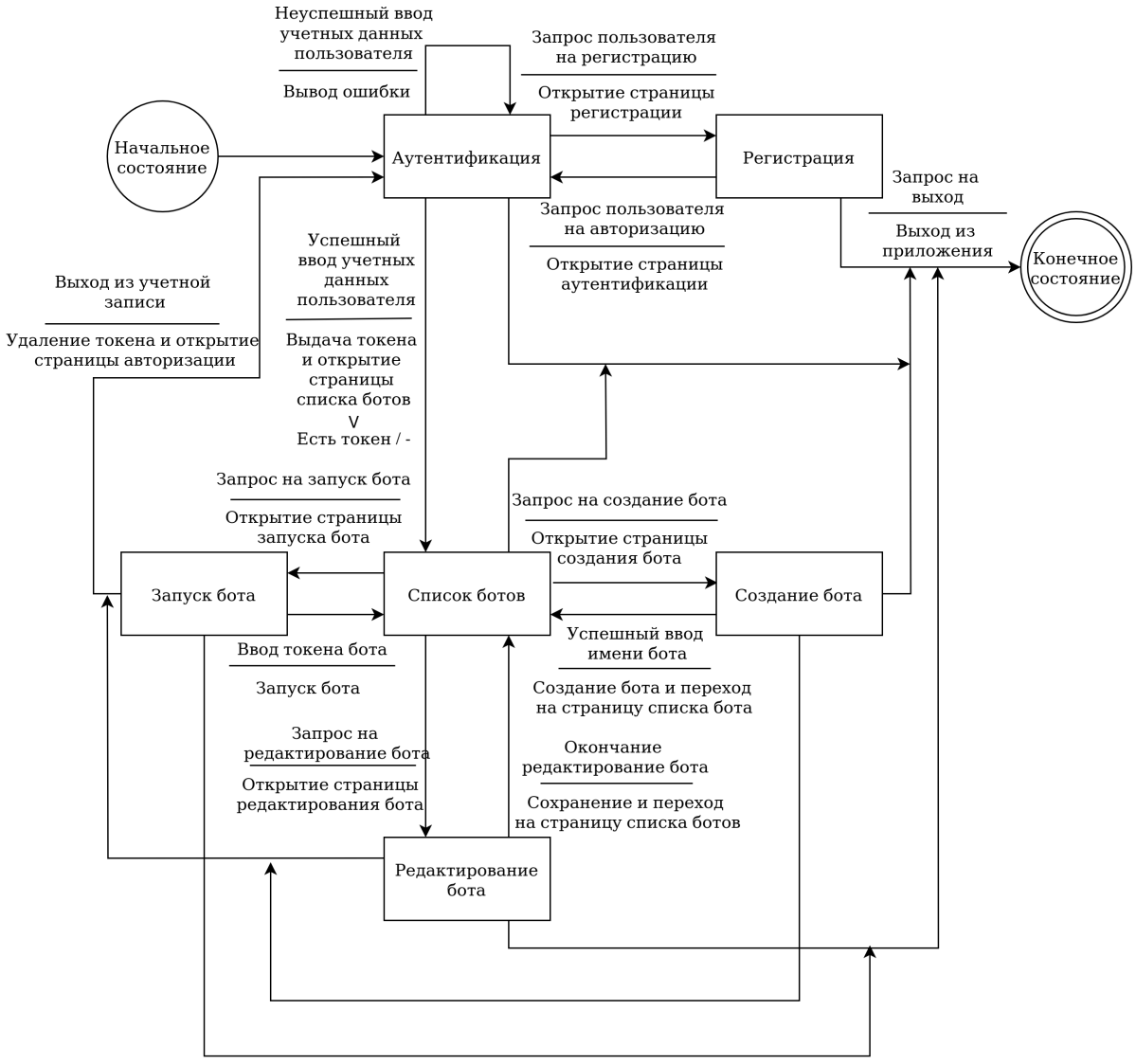

Рисунок 13 – Диаграмма состояний клиентской части конструктора

При запуске приложения пользователя ожидает страница входа в систему.

Если пользователь не зарегистрирован, то ему предлагается перейти на страницу регистрации.

При правильном вводе логина и пароля пользователю предоставляется токен доступа, и открывается страница списка ботов, иначе – выводится ошибка.

На странице списка ботов пользователю предоставляется выбрать бота для редактирования, удаления или запуска/остановки. Также пользователь может перейти на страницу создания бота.

На странице создания пользователь может ввести название бота, и после нажатия на кнопку “создать” происходит создание бота с последующим возвратом на страницу списка ботов. Список ботов обновляется – в списке уже присутствует вновь созданный бот.

При запуске бота отрывается страница с вводом токена бота. Если токен был введен успешно, то происходит запуск бота с возвратом на страницу списка ботов.

При нажатии на элемент списка ботов происходит открытие редактора, где пользователь может модифицировать бота: создавать и изменять компоненты, соединять их.

При закрытии любой страницы происходит выход из приложения, также при попытке получения доступа к любой из страниц без валидного токена происходит переход на страницу входа в приложение.

3.3.3.2 Диаграмма состояний визуального редактора

Визуальный редактор также имеет конечный набор состояний. Диаграмма состояний визуального редактора представлена на рисунке 14.

В начале своего запуска визуальный редактор ожидает действия от пользователя, в зависимости от которых происходит изменение его состояния.

В редакторе присутствует область выбора компонента. Если пользователь решит выбрать компонент – инициируется переход в состояние добавления компонента.

В момент перехода в состояние добавления происходит создание временного компонента, после чего пользователь может его перемещать. Если конечное

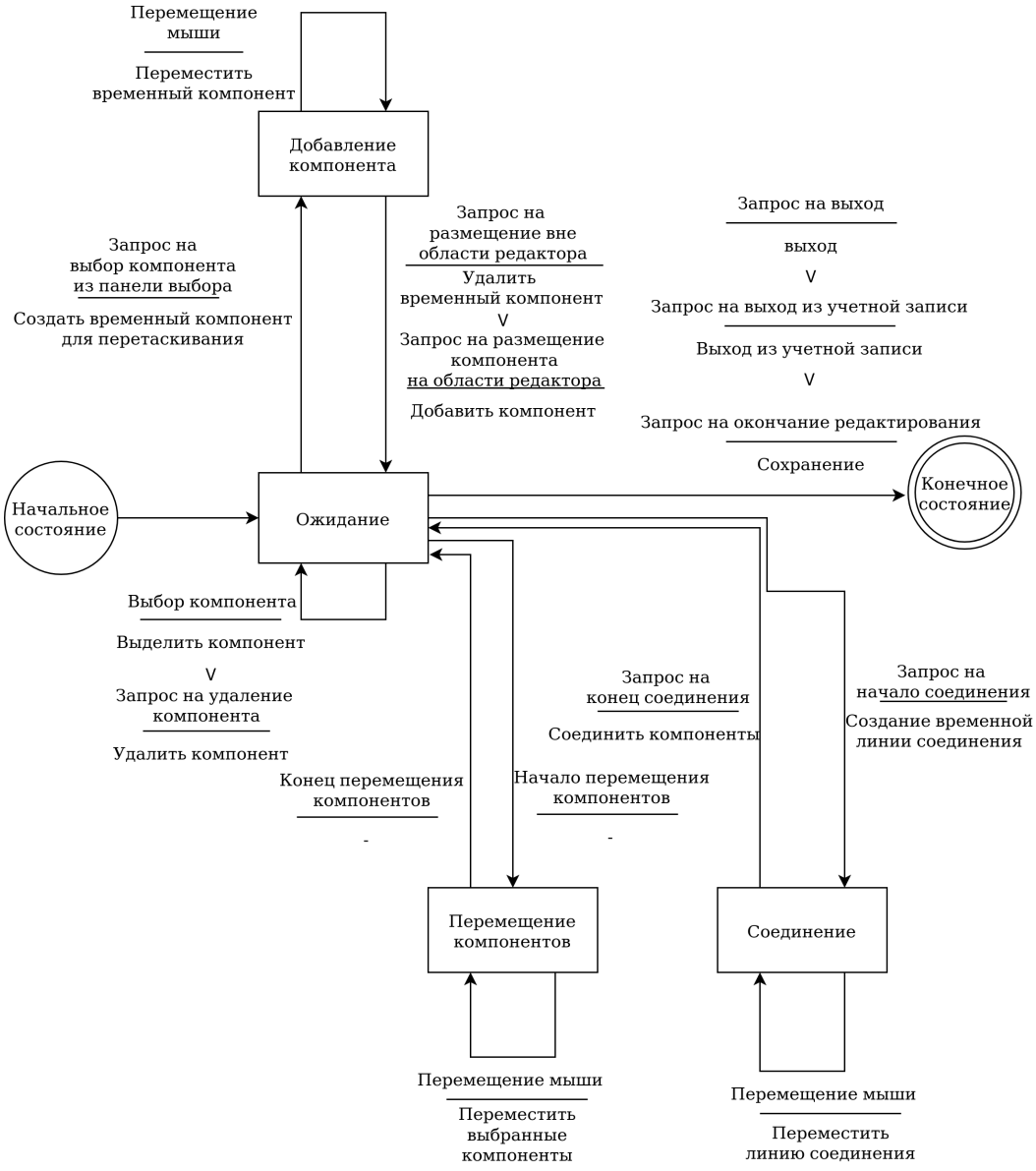

Рисунок 14 – Диаграмма состояний визуального редактора

расположение компонента выбрано – инициируется переход обратно в состояние ожидания, при этом временный компонент заменяется постоянным.

Нажимая на любую выходную точку компонента, пользователь дает запрос редактору на переход в режим соединения компонентов, следствием чего является появление линии, которая следует за курсором мыши. Также при этом у всех других компонентов, кроме начального появляется область соединения при наведении мыши на неё. При отпускании левой кнопки мыши на этой области происходит закрепление линии – признак соединения компонентов. Если же отжатие кнопки мыши происходит вне этой области, то линия теряется и соединения не

происходит. В любом из этих случаев происходит переход в состояние ожидания.

Отсоединение компонентов происходит при нажатии левой кнопки мыши на входную точку одного из компонентов, и при этом редактор переходит в состояние перемещении линии – состояния соединения компонентов.

При выборе компонента появляется возможность его перемещения. Если пользователь при уже нажатой левой кнопки мыши на компоненте начнет её перемещать, то компонент начнёт перемещение за ней – состояние перемещения. Отпускание левой кнопки мыши закрепит компонент, и редактор снова начнет ожидать действия от пользователя.

3.3.4 Расчёт координат объектов визуального редактора

Область редактора представляет собой координатную плоскость, на которой располагаются компоненты бота. Расположение компонента обеспечивается координатами (x_c, y_c), которые указывают на левый верхний угол компонента.

При перемещении компонента вычисляются смещения $\triangle x$ и $\triangle y$ относительно координат нажатой мыши (x_m, y_m) по формулам

$$\triangle x = x_c - x_m, \tag{1}$$

$$\triangle y = y_c - y_m. \tag{2}$$

Данные смещения используются для расчёта новых координат компонента (x'_c, y'_c) при перемещении мыши с координатами (x'_m, y'_m), которые вычисляются по формулам

$$x'_c = x'_m - \triangle x, \tag{3}$$

$$y'_c = y'_m - \triangle y. \tag{4}$$

Расположение компонента на координатной плоскости показано на рисунке 15.

Связи между компонентами представляют собой линию со стрелкой. Линия имеет координаты начала (x_{out}, y_{out}) и конца (x_{in}, y_{in}), которые представляют собой точки центра окружностей соединительных точек выхода и входа компо-

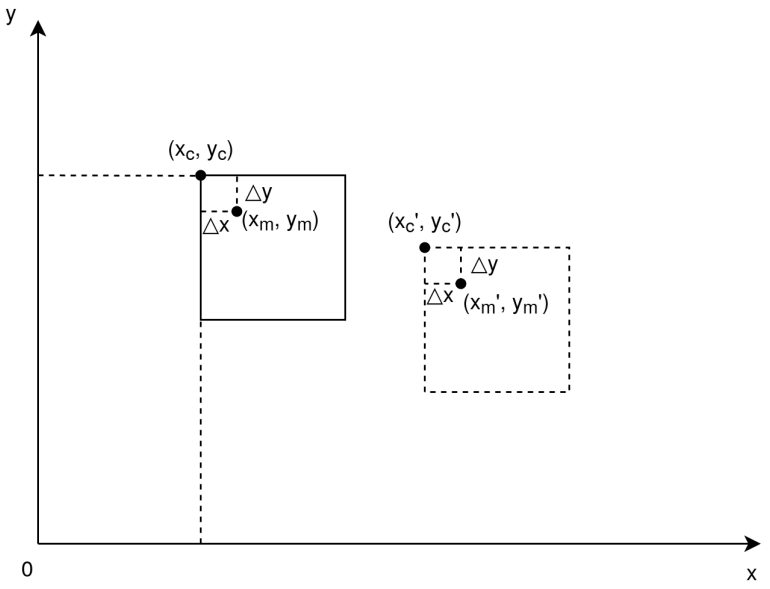

Рисунок 15 – Координаты расположения компонентов

нентов. Расположение связей компонентов на координатной плоскости показано на рисунке 16.

Стрелка представляет собой две примыкающих к линии прямых. Стрелка имеет длину a и угол между примыкающих прямых α.

Линия между компонентами наклонена под углом β относительно оси y. Угол вычисляется по формуле

$$\beta = \text{arctan}(\frac{x_{in} - x_{out}}{y_{in} - y_{out}}). \quad (5)$$

Точка (x_a, y_a) является окончанием стрелки и её координаты вычисляются по формулам

$$x_a = x_{in} - sin\beta * a, \quad (6)$$

$$y_a = y_{in} - cos\beta * a. \quad (7)$$

Расчет смещения b примыкающих прямых относительно основной линии и

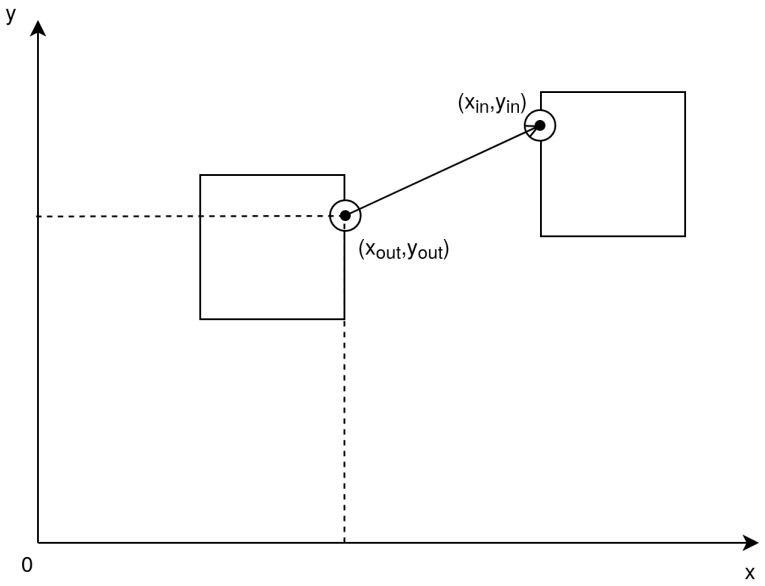

Рисунок 16 – Координаты расположения связей компонентов

смещений $\triangle x_a$ и $\triangle y_a$ относительно осей x и y происходит по формулам

$$b = \tan(\frac{\alpha}{2}) * a, \tag{8}$$

$$\triangle x_a = \cos(\beta) * b, \tag{9}$$

$$\triangle y_a = \sin\beta * b. \tag{10}$$

Сами точки окончания примыкающих прямых (x_{a1}, y_{a1}) и (x_{a2}, y_{a2}) вычисляются по формулам

$$x_{a1} = x_a + \triangle x_a, \tag{11}$$

$$y_{a1} = y_a - \triangle y_a, \tag{12}$$

$$x_{a2} = x_a - \triangle x_a, \tag{13}$$

$$y_{a2} = y_a + \triangle y_a. \tag{14}$$

Расположение стрелки на координатной плоскости представлено на рисунке 17.

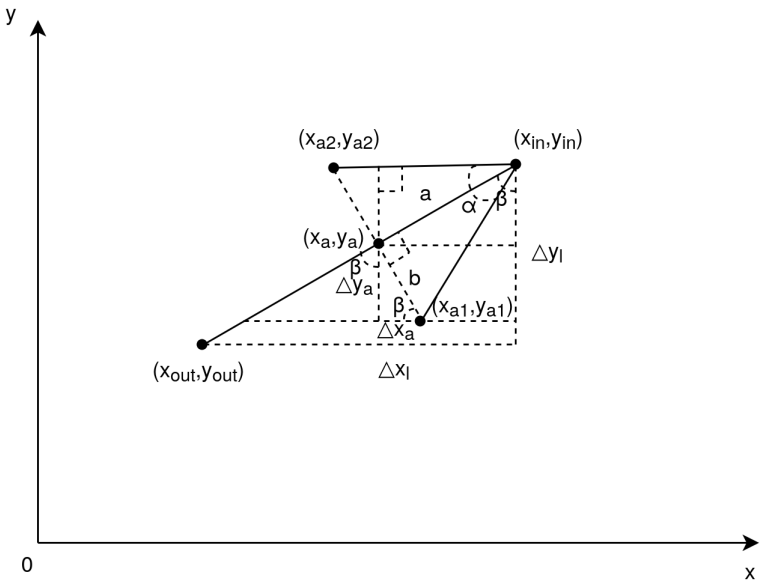

Рисунок 17 – Координаты расположения стрелки

Выводы

В данном разделе была определена визуальная структура серверной и клиентской частей конструктора.

Модульная структура серверной части и визуального редактора позволила выделить определенные функциональные блоки – набор структур и функций, которые ответственны за определенную часть системы. Компонентная структура помогла представить редактор как набор связанных компонентов в виде дерева, эти компоненты взаимодействуют друг с другом.

Были выделены алгоритмы функционирования серверной части конструктора, такие как обработка запросов пользователей сервиса ботов и пользователей ботов. Также было определено поведение клиентской части и входящего в него визуального редактора с помощью диаграммы состояний.