МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»**
Кафедра дискретной математики и информационных технологий

РАЗРАБОТКА ВИРТУАЛЬНОГО ПЕРСОНАЛЬНОГО АССИСТЕНТА С ИСПОЛЬЗОВАНИЕМ ФРЕЙМВОРКА VERT.X

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 421 группы
направления 09.03.01 — Информатика и вычислительная техника
факультета КНиИТ
Згонникова Даниила Евгеньевича

Научный руководитель
доцент, к. ф.-м. н. _____ Л. Б. Тяпаев

Заведующий кафедрой
доцент, к. ф.-м. н. _____ Л. Б. Тяпаев

Саратов 2020

ВВЕДЕНИЕ

Сегодня мир становится все более мобильным и интернет-ориентированным. Многие компании переходят в «виртуальный канал общения» со своими клиентами. И рано или поздно приходят к необходимости выбора средств для эффективного взаимодействия с клиентами. Сегодняшние информационный рынок предлагает два основных средства такого взаимодействия: чат-бот или собственное мобильное приложение с функциями обратной связи. Поэтому в этих условиях большую актуальность имеет определение сферы и направлений, когда целесообразнее использовать каждый из данных продуктов.

Чат-бот – это специальная программа, предоставляющая пользователю возможность имитирования реального разговора. Он стал популярным форматом взаимодействия с пользователями в соцсетях, поэтому всё больше компаний задумываются создать своего чат-бота. Чат-боты работают на разных платформах: Фейсбук Мессенджер, Телеграм, ВКонтакте, Вайбер, Слак, Скайп и т.д. Чат-боты становятся важным, полезным, необходимым инструментом современной жизни. С точки зрения бизнеса чат-боты уменьшают нагрузку на call-центры, снижают стоимость обслуживания и увеличивают выручку.

Целью работы является описание решения по созданию Виртуального персонального помощника (VPA). Система предназначена для автоматизации взаимодействия сотрудников группы компаний “ГМК Норильский Никель” с сервисами корпоративных систем, используя приложение для обмена мгновенными сообщениями Viber.

Поставленная цель определила следующие задачи:

1. Анализ основных видов реализации чат-ботов;
2. Изучение методов и инструментов для разработки приложения;
3. Разработка архитектурного решения для приложения;
4. Разработка сценариев взаимодействия приложения;

Бакалаврская работа состоит из введения, 4 разделов, заключения, списка использованных источников и 0 приложений. Общий объем работы - 43 страницы, из них 40 страницы - основное содержание, включая 11 рисунков и 3 таблицы, список использованных источников информации - 20 наименований.

КРАТКОЕ СОДЕРЖАНИЕ РАБОТЫ

Первый раздел **«Предметная область»** посвящен рассмотрению основных типов чат-ботов и разбору программных инструментов для их разработки.

Всего существует два вида реализации чат-ботов:

1. Обучающиеся боты (понимают естественные языки), которые используют логику при построении диалога или обработку естественного языка (Natural Language Processor) и машинное обучение (Machine Learning) для формирования ответов на сообщения.
2. Скриптовые боты (не понимают естественные языки), в которых весь диалог - это заранее сформированный шаблон, а скрипт - это дерево решений, в котором ответ на вопрос открывает новый, заранее запрограммированный сценарий. Диалог в них обычно линейны и структурированы.

В работе представлена разработка скриптового чат-бота, так как он проще в освоении, а использование сценариев позволяет дополнять функционал бота без потребности в изменении основного ядра приложения.

Для построения серверной части приложения был выбран язык Java с использованием библиотек Vert.x, JSON.org и Viber API.

Vert.x — это событийно-ориентированный инструментарий, поддерживающий несколько языков и работающий на виртуальной машине Java.

Ядро Vert.x является обычной библиотекой jar, поэтому он может быть встроен в приложения, упакованные в виде набора jar-архивов, одного

jar-архива со всеми зависимостями или даже может быть развернут внутри популярных контейнеров компонентов и приложений.

Приложения Vert.x состоят из компонентов (verticles). Каждый компонент управляется событиями, то есть компоненты не запускаются до тех пор, пока не получат сообщение. Компоненты могут связываться друг с другом через шину событий.

Шина событий (event bus) — основной инструмент для связи компонентов через асинхронную передачу сообщений. Она позволяет передавать любые данные (сообщения могут быть простыми объектами, строками, данными в формате CSV, JSON, двоичными данными и т. д.), но предпочтительным форматом обмена является JSON, так как он позволяет связывать компоненты, написанные на разных языках. Шина событий поддерживает следующие шаблоны связи:

1) обмен сообщениями от одной точки к другой (point-to-point messaging),
2) обмен сообщениями «запрос - ответ»,
3) подписка / отписка на трансляцию сообщений.

Компоненты Vert.x обрабатывают входящие события в цикле, где события могут быть похожими на прием сетевых буферов, событий синхронизации или сообщений, отправленных другими компонентами.

Циклы событий (event loops) типичны для асинхронных моделей программирования. Цикл принимает на вход очередь событий и отправляет каждое событие его обработчику, таким образом, цикл событий может доставлять огромное количество сообщений за короткий промежуток времени. Этот паттерн называется реактор (reactor pattern).

Vert.x работает по-другому: вместо одного цикла событий каждый экземпляр Vert.x поддерживает несколько циклов событий (по умолчанию

берется число, основанное на количестве доступных ядер на машине), из чего следует, что один процесс Vert.x может масштабироваться на сервере. Этот паттерн называется мультиреакторным (multireactor pattern).

Общение между системами в приложении происходит посредством HTTPS запросов.

Протокол Передачи Гипертекста (HTTP), один из протоколов стека TCP/IP (Transmission Control Protocol/Internet Protocol), был изначально разработан для публикации и получения HTML (Hyper Text Markup Language) страниц и теперь используется для распределенных информационных систем. HTTP используется во Всемирной Паутине для передачи данных и является одним из самых широко применяемых прикладных протоколов.

HTTP определяет протокол типа запрос/ответ. Когда клиент, например, веб браузер, посылает сообщение запроса на сервер, HTTP протокол определяет типы сообщений, используемые клиентом для запроса веб страницы, а также типы сообщений, применяемые сервером для ответа. Тремя распространёнными типами сообщений являются GET, POST и PUT.

POST и PUT используются, чтобы посылать сообщения, которые загружают данные на веб сервер. Например, когда пользователь вводит данные в форму, встроенную в веб страницу, POST включает данные в сообщение, посылаемое на сервер. PUT загружает ресурсы или контент на веб сервер.

JSON (JavaScript Object Notation) - простой формат обмена данными, удобный для чтения и написания как человеком, так и компьютером. Он основан на подмножестве языка программирования JavaScript, определенного в стандарте ECMA-262 3rd Edition - December 1999. JSON - текстовый формат, полностью независимый от языка реализации, но он использует соглашения, знакомые программистам C-подобных языков, таких как C, C++, C#, Java, JavaScript, Perl, Python и многих других. Эти свойства делают JSON идеальным языком обмена данными.

В основу работы Viber REST API заложена обработка Вебхуков и использования токена аутентификации.

Токен аутентификации (также известный как “ключ приложения”) является уникальным и секретным идентификатором учетной записи. Он используется для предотвращения отправки посторонними лицами запросов от имени бота. Каждый запрос к API должен содержать заголовок HTTP с именем X-Viber-Auth- Token, содержащий токен аутентификации учетной записи [16].

Вебхуком называют функционал уведомления сторонних сервисов. Это означает, что если в чат приходит сообщение, то Viber сам сообщает сервису об этом событии. Таким образом отпадает необходимость периодически опрашивать сервера, тем самым, исчезает причина “падений” ботов. Однако за подобный функционал приходится платить необходимостью установки полноценного веб-сервера на машину, где планируется развертывать ботов. Также для работы потребуется наличие собственного SSL-сертификата (Secure Sockets Layer), так как вебхуки в Viber работают только по протоколу HTTPS.

Во втором разделе **«Архитектура приложения»** рассматривается внутреннее строение приложения.

Платформа VPA состоит из следующих компонентов:

1. VPA Bot - автоматизированный аккаунт (бот) в приложении Viber.
2. VPA Server - серверная часть VPA.
3. Клиентские системы - авторизованные системы ГК Норникель, осуществляющие обращение к базе данных сотрудников ГК Норникель, внутренним системам отдела кадров компании и т.д. (в текущем решении Автоматическая Система Управления Персоналом - единственная интегрированная система-клиент).

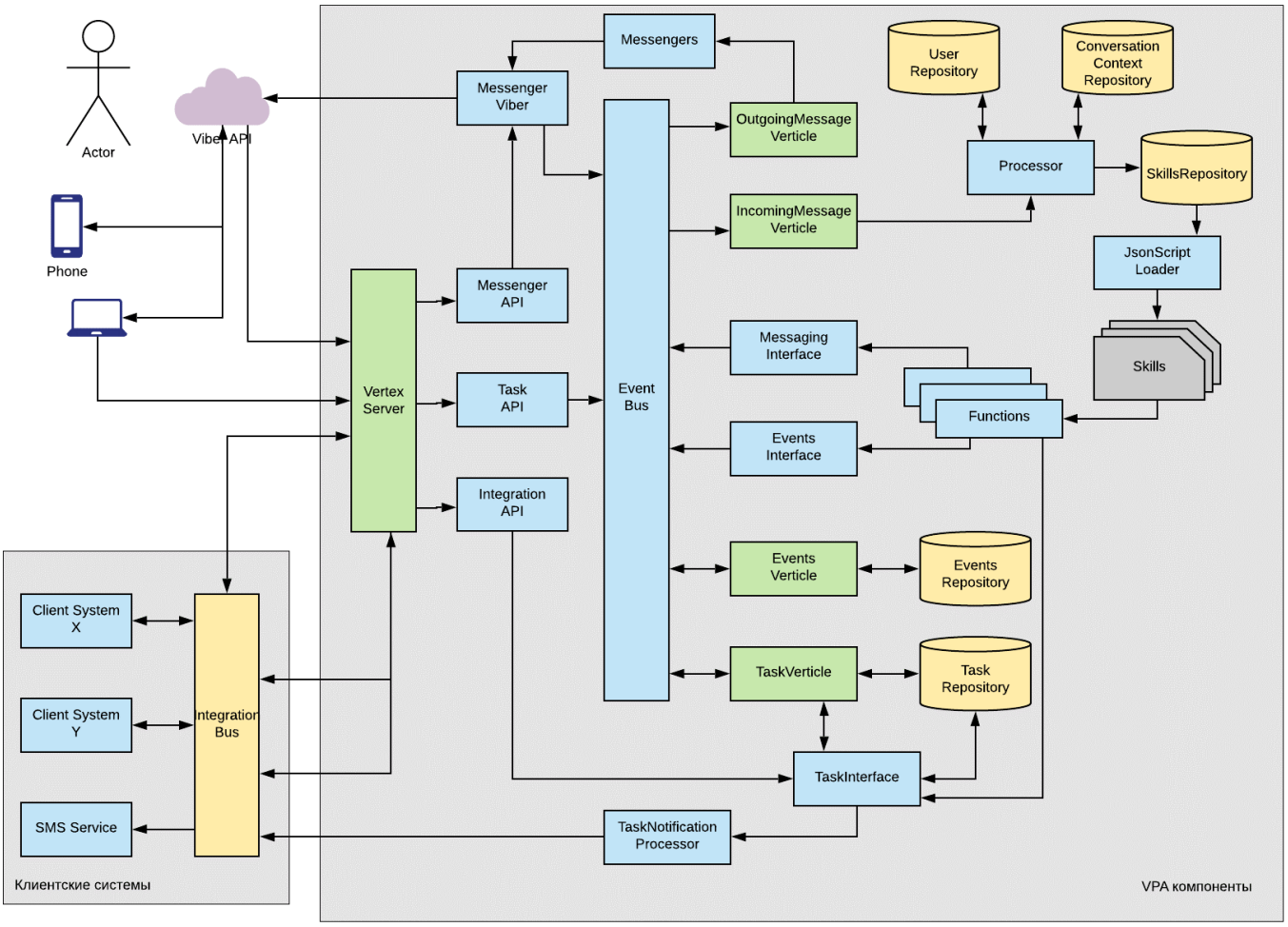

Рисунок 1 - Архитектурная диаграмма VPA

Из приведенной на рисунке 1 диаграммы можно выделить следующие основные элементы:

- Viber API - Набор готовых классов, методов и функций, предоставляемых мессенджером Viber по протоколу https, позволяющий автоматизировать аккаунт (создать бота) для общения с пользователями.
- VertxServer - Класс VPA, отвечающий за обработку входящих http-запросов и осуществляющий их маршрутизацию в соответствующие модули управления внутри платформы.

- Event Bus - Шина событий является связующим элементом, обеспечивающим унифицированный событийно-ориентированный обмен сообщениями между различными компонентами системы VPA
- IncomingMessageVerticle - Класс-обработчик входящих сообщений, работающий в отдельном потоке и подписанный на события в шине EventBus “vpa.messaging.incomingMessages”. При появлении в очереди шины события соответствующего типа извлекает его из очереди и передает их для процессинга компоненту Processor.
- OuntgoingMessageVerticle - Компонент-обработчик исходящих сообщений, работающий в отдельном потоке и подписанный на события в шине EventBus “vpa.messaging.outgoingMessages”. При появлении в очереди шины события соответствующего типа извлекает его из очереди и передает для преобразования и отправки конечному пользователю в компонент Messenger.
- EventsVerticle - Компонент, отвечающий за обработку событий для логирования, работающий в отдельном потоке и подписанный на события в шине EventBus “vpa.events.create”.
- TasksVerticle - Класс-обработчик задач, работающий в отдельном потоке и подписанный на события в шине EventBus “vpa.tasks.create”, “vpa.tasks.update” и “vpa.tasks.poll”. При появлении в очереди шины события соответствующего типа извлекает его из очереди, производит необходимые изменения, обращаясь к TaskRepository (например, находит для входящего обновления по задаче исходную задачу, обновляет статус задачи), и передает их для процессинга через шину EventBus в очередь “vpa.tasks.post_update” для дальнейшей обработки в IncominMessageVerticle. Также компонент запускает периодическую процедуру удаления задач из хранилища и периодический запуск запланированных задач для сконфигурированных очередей.

- Processor - Компонент, отвечающий за общую логику процессинга сообщений, задач и обновлений по задачам. В случае процессинга сообщения: определяет пользователя-отправителя, контекст сообщения и навык, отвечающий за бизнес-сценарии обработки сообщения. В случае процессинга задачи: определяет пользователя-получателя, его контекст выполнения и навык/набор навыков, содержащие лисенеры для обработки задачи. В случае процессинга обновления по задаче: определяет исходную задачу, пользователя-получателя, его контекст выполнения и навык/набор навыков, содержащие листенеры для обработки обновления по задаче.
- Skill - Расширяемый набор сценариев выполнения взаимодействия VPA как с интегрируемыми системами, так и с пользователями в мессенджере. Представляет собой набор json-файлов, подгружаемых в приложение VPA.

В третьем разделе **«Сценарии взаимодействия»** рассматривается подробная работа компонентов программы при трех различных ситуация:

1) Отправка сообщения конечным пользователем через установленное на смартфоне/компьютере приложение Viber
2) Запрос клиентской системы списка задач у VPA
3) Отправка уведомления информационной системой пользователю Viber.

В четвертом разделе **«Описание интерфейса чат-бота»** рассмотрена работа чат-бота на примере выполнения сценария “Запланированные отпуска.

ЗАКЛЮЧЕНИЕ

Эффективная работа крупной корпорации во многом зависит от того, как быстро можно найти ответ на нужный вопрос в соседнем отделе или департаменте, как можно быстро получить доступ к различным корпоративным сервисам, также получить необходимые справки для личных нужд.

Таким образом, разработанное приложение Виртуального Персонального Ассистента предоставило легкий и оперативный доступ сотрудникам к информации и необходимым функциям вместо ожидания ответа на e-mail, звонок через кадровый портал и дополнительные приложения.

Реализация программного модуля, отвечающего за выполнение бизнес-сценариев, определенных в рамках проектного решения, а также аналогичных сценариев предусматривает возможность описать бизнес-сценарий в виде конфигураций в формате JSON без необходимости добавлять программный код на Java. Подобный подход позволяет масштабировать решение путем добавления новых бизнес-сценариев без дополнительной разработки программных модулей.

ОСНОВНЫЕ ИСТОЧНИКИ ИНФОРМАЦИИ

1. Документация Vert.x [Электронный ресурс]: URL: http://vertx.io/docs/vertx-core/java/ (Дата обращения: 23.04.2020)
2. Escoffer C. Building Reactive Microservices in Java [Электронный ресурс] / C. Escoffer. — Электрон. дан. — O'Reilly Media, 2017. (Дата обращения: 23.04.2020)
3. A gentle guide to asynchronous programming with Eclipse Vert.x for Java developers. [Электронный ресурс]: URL: http://vertx.io/docs/guide-for-java-devs/
4. Чат-боты: где, как и когда заменить человека [Электронный ресурс] – URL: https://geektimes.ru/company/asus/blog/289997/ (Дата обращения: 21.05.2020).
5. Что такое чат-бот? [Электронный ресурс]; – URL: https://jetstyle.ru/services/bots (Дата обращения: 21.05.2020).