Министерство образования и науки Российской Федерации

ФГБОУ ВО «СИБИРСКИЙ ГОСУДАРСТВЕННЫЙ

УНИВЕРСИТЕТ НАУКИ И ТЕХНОЛОГИЙ ИМЕНИ АКАДЕМИКА М.Ф.РЕШЕТНЕВА» (СибГУ)

Институт информатики и телекоммуникаций

Кафедра информационно-управляющих систем

ДОПУСТИТЬ К ЗАЩИТЕ В ГЭК

Зав. кафедрой

д-р. техн. наук, профессор

\_\_\_\_\_\_\_\_\_\_\_\_ А.В.Мурыгин

(подпись)

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2017г.

Хакимов Дамир Рашитович

**РАЗРАБОТКА СИСТЕМЫ ПОДДЕРЖКИ ПРИНЯТИЯ РЕШЕНИЙ В ЕЖЕДНЕВНОМ БЫТУ**

(ИУС.00.00.00.138.ПЗ)

**МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ**

на соискание степени магистра

по направлению подготовки

09.04.04 – «Программная инженерия»

Канд.техн.наук, доцент

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.С.Коморовский

(подпись, дата)

Док.эконом.наук,

вед.науч.сот ИЭОПП

СО РАН

Рецензент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Г.И. Поподько

(подпись, дата)

Доцент кафедры ИУС

Нормоконтроль \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ С.С.Москалева

(подпись, дата)

Автор работы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Д.Р.Хакимов

(подпись, дата)

Красноярск, 2017

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

**«Сибирский государственный университет науки и технологий имени академика М.Ф. Решетнева»**

**(СибГУ)**

УТВЕРЖДАЮ

Зав. кафедрой ИУС

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_\_г.

**ЗАДАНИЕ**

**на магистерскую диссертацию**

Студенту\_\_\_\_\_\_\_Хакимову Дамиру Рашитовичу группы 22-7м\_\_\_\_\_\_

направления\_\_\_ «Программная инженерия»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

магистерской программы \_ 09.04.04 – «Программная инженерия»\_\_\_\_\_\_\_\_\_\_\_\_

Тема диссертации \_Разработка системы поддержки принятия решений в ежедневном быту\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Утверждена приказом по университету №\_\_\_\_\_\_\_от «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_20\_\_\_\_г.

Срок сдачи студентом первого варианта диссертации «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_20\_\_\_\_г.

Срок сдачи студентом законченной диссертации «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_\_г.

Исходные данные к диссертации\_Юн, С. Г. Система поддержки принятия управленческих решений на основе совершенствования технологий накопления и хранения данных: на примере промышленных предприятий : дис. канд. тех. наук:\_05.13.10 / Юн, Светлана Геннадьевна – Новосибирск, 2007 – 189 с. \_\_ \_\_Абдулин, Е. Р. Методы проектирования программного обеспечения и человеко-машинного взаимодействия в многофункциональных мультимедийных комплексах : дис. канд. тех. наук: 05.13.11 / Абдулин, Евгений Рудольфович – Москва, 2011 – 173 с. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Содержание диссертации (перечень вопросов, подлежащих разработке)  
\_Провести глубокий анализ систем поддержки принятия решений и рынка BI-систем; изучить принцип работы публичных API; написание алгоритма адаптации информационных технологий систем поддержки принятия решений к современным каналам человеко-машинного взаимодействия в виде бота-информера; создать блок-схему на основе написанного алгоритма; разработать систему поддержки принятия решений в новых каналах человека машинного взаимодействия в виде бота для мессенджера telegram.\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Перечень графического материала с указанием обязательных чертежей

\_\_Блок-схема программной реализации бота-ифнормера для мессенджера telegram\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Нормоконтроль\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата выдачи задания «\_\_\_»\_\_\_\_\_\_ 20\_\_г. \_\_\_\_\_\_\_\_\_\_ /Коморовский В.С./

**АННОТАЦИЯ**

ХАКИМОВ ДАМИР РАШИТОВИЧ

РАЗРАБОТКА СИСТЕМЫ ПОДДЕРЖКИ ПРИНЯТИЯ РЕШЕНИЙ В ЕЖЕДНЕВНОМ БЫТУ

НАПРАВЛЕНИЕ \_\_«Программная инженерия»\_\_\_\_\_\_\_\_\_\_

МАГИСТЕРСКАЯ ПРОГРАММА \_\_09.04.04. «Программная инженерия» \_

УЧЕБНАЯ ГРУППА \_\_22-7М\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ГОД ЗАЩИТЫ \_\_2017 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ПРИМЕР.** В данной магистерской диссертации рассмотрен вопрос о создании, развитии и внедрении системы поддержки принятия решения в повседневную жизнь человека, посредством бота-информера в мессенджере Telegram. Проанализирован рынок СППР. Выделены и описаны преимущества и недостатки системы, исследовано применение публичных API как одного из помощников в реализации СППР. Описаны рекомендательные сервисы, область применения, список рекомендательных сервисов. Выбрана система управления базами данных, среда разработки посредством сравнения. Выбрана и обоснована программа для реализации внедряемого бота-информера.

**THE SUMMARY**

DEVELOPMENT OF THE SYSTEM OF SUPPORT OF DECISION-MAKING DURING EVERYDAY LIFE.

**EXAMPLE.** In this master's thesis, the question of creating, developing and implementing a decision support system in the everyday life of a person, through a bot-informer in the Telegram messenger, was considered. Analyzed market DSSR. The advantages and disadvantages of the system are singled out and described, the application of public APIs as one of the assistants in the implementation of the DSS is explored. Descriptive services, scope of application, list of recommended services are described. A database management system has been chosen, the development environment through comparison. The program for implementation of the introduced informer bot has been selected and justified.

Содержание

[Введение 4](#_Toc485278855)

[1 Системы поддержки принятия решений 6](#_Toc485278856)

[1.1 Определение систем поддержки принятия решений и их применение 6](#_Toc485278857)

[1.2 История развития систем поддержки принятия решений 15](#_Toc485278858)

[1.3 Метод анализа иерархий 17](#_Toc485278859)

[1.4 Использование и применение МАИ 19](#_Toc485278860)

[1.5 Образование и научные исследования МАИ 21](#_Toc485278861)

[1.6 Пример задачи многокритериального выбора с простейшей иерархией МАИ 22](#_Toc485278862)

[1.7 Сфера образования и научных исследований МАИ 23](#_Toc485278863)

[1.8 Методика применения МАИ 24](#_Toc485278864)

[1.9 Рынок СППР 25](#_Toc485278865)

[1.10 Состояние Российского рынка Business Intelligence 26](#_Toc485278866)

[2 Публичные API 28](#_Toc485278867)

[2.1 API один из новых каналов взаимодействия 28](#_Toc485278868)

[2.2 История API 31](#_Toc485278869)

[2.3 Применение публичных API 31](#_Toc485278870)

[3 Рекомендационные сервисы 36](#_Toc485278871)

[3.1 Определения и применение рекомендационных сервисов 36](#_Toc485278872)

[3.2 Список рекомендательных сервисов 38](#_Toc485278873)

[4 Выбор системы управления базами данных 43](#_Toc485278874)

[4.1 История PostgreSQL 43](#_Toc485278875)

[4.2 PostgreSQL 46](#_Toc485278876)

[4.3 Сравнение PostgreSQL c другими СУБД 48](#_Toc485278877)

[5 Среда разработки. 53](#_Toc485278878)

[5.1 Причины использования Python 54](#_Toc485278879)

[5.2 История создания 57](#_Toc485278880)

[5.3 Влияние других языков 57](#_Toc485278881)

[5.4 Преимущество Python перед другими языками высокого уровня 58](#_Toc485278882)

[5.5 Реализации Python 59](#_Toc485278883)

[5.6 Недостатки 60](#_Toc485278884)

[6 Мессенджер Telegram 62](#_Toc485278885)

[6.1 Основные причины выбора данного мессенджера 62](#_Toc485278886)

[6.2 История 63](#_Toc485278887)

[6.3 Технология 66](#_Toc485278888)

[6.4 Боты 66](#_Toc485278889)

[6.5 Преимущества и недостатки использования Telegram 67](#_Toc485278890)

[6.6 Секретный чат 67](#_Toc485278891)

[6.7 Мессенджеры в России 68](#_Toc485278892)

[7 Моделирование системы поддержки принятия решений при обращении к боту мессенджеру 72](#_Toc485278893)

[Заключение 77](#_Toc485278894)

[Библиографический список 79](#_Toc485278895)

[Приложение А 83](#_Toc485278896)

[Приложение Б 86](#_Toc485278897)

Введение

**Актуальность исследования.** Первые СППР появились в начале семидесятых годов, и требовали мощных ресурсов для работы и реализации. Теперь СППР используются в различных отраслях, например, в медицине, в бизнесе, а также помогают в выборе специалистов на определённую должность и, что самое главное, теперь абсолютно каждый может использовать эту систему в своих целях.

Существенное увеличение технических и технологических мощностей позволяет сейчас создавать СППР на базе готовых библиотек и компонентов. По оценкам сайта [www.tadviser.ru](http://www.tadviser.ru) объем российского рынка BI-систем (ранее СППР) в 2017 составит приблизительно 55 миллиардов рублей. Тенденция роста на 5-10 процентов каждый последующий год обуславливает актуальность и целесообразность исследования данного направления.

В век информатизации гаджеты вошли в нашу жизнь и остаются очень нужными и удобными вещами, а смартфоны приобретают всё более мощные характеристики и используются как замена в иной раз компьютеров. Развиваются мобильные операционные системы и для каждой создаются мобильные приложения, для различного рода деятельности или развлечения. одно из таких приложений telegram. Это приложение представляет собой оболочку для общения клиента с клиентом по защищённому соединению, далее оно приобрело ещё одну функцию и такую как бот-информер, с помощью таких ботов можно упростить некоторые рутинные функции в ежедневном быту.

Исходя из актуальности, выявлена проблема адаптации информационных технологий систем поддержки принятия решений к современным каналам человеко-машинного взаимодействия.

**Цель исследования.** Создание системы поддержки принятия решений с интерфейсом в виде бота для мессенджера telegram.

**Задачи:**

* анализ систем принятия решений и методы их применения;
* анализ и описание применения публичных API;
* создание базы данных в СУБД;
* написание алгоритма работы СППР к каналам человеко-машинного взаимодействия в виде бота для мессенджера telegram;
* интегрирование БД со средой программирования;
* создание блок-схемы;
* написание бота для мессенджера telegram.

**Научная новизна исследования.** Создание канала человеко-машинного взаимодействия в виде бота мессенджера для telegram, используя данные системы поддержки принятия решений в бытовых целях.

**Личный вклад автора.** В работе проведён глубокий анализ систем поддержки принятия решений и рекомендательных сервисов. Дано обоснования применения данной системы для реализации в виде нового канала человеко-машинного взаимодействия. Разработан алгоритм работы данной системы и собрана база данных запросов пользователей.

**Публикации.**

1. Жудрак А.П., Хакимов Д.Р. Применение имитационного моделирования для улучшения качества жилищно-коммунальных услуг // 52-ая Международная научная студенческая конференция "Студент и научно-технический прогресс". МНСК сборник тезисов 2014 – Новосибирск, 2014. С. 242.

2. Жудрак А.П., Хакимов Д.Р. Применение имитационного моделирования для улучшения системы обслуживания на предприятиях ЖКХ // молодые ученые в решении актуальных проблем науки: ВНПК. Сборник статей студентов, аспирантов и молодых ученых. Том III. – Красноярск, 2015. С. 179-181.

3. Хакимов Д.Р. Применение в учебном процессе ментальных карт // Образовательные ресурсы и технологии. Выпуск 2016'1(13). Московский университет им. С.Ю. Витте – Москва, 2016. С. 3-8.

1 Системы поддержки принятия решений

1.1 Определение систем поддержки принятия решений и их применение

Термин «система поддержки принятия решений» появился в начале семидесятых годов. За это время дано много определений СППР. Так в она определяется следующим образом: «Системы поддержки принятия решений являются человеко-машинными объектами, которые позволяют лицам, принимающим решения (ЛПР), использовать данные, знания, объективные и субъективные модели для анализа и решения слабоструктурированных и неструктурированных проблем» В этом определении подчеркивается предназначение СППР для решения слабоструктурированных и неструктурированных задач.

В той или иной степени, системы поддержки принятия решений (СППР) присутствуют в любой информационной системе (ИС). Поэтому к задаче создания системы поддержки принятия решений организации приступают сразу после приобретения вычислительной техники и установки программного обеспечения. По мере развития бизнеса, упорядочения структуры организации и налаживания межкорпоративных связей, проблема разработки и внедрения СППР становится особенно актуальной. Одним из подходов к созданию таких систем стало использование хранилищ данных.

СППР можно, в зависимости от данных, c которыми они работают, разделить на оперативные, предназначенные для немедленного реагирования на текущую ситуацию, и стратегические - основанные на анализе большого количества информации из разных источников с привлечением сведений, содержащихся в системах, аккумулирующих опыт решения проблем.

В соответствии с к слабоструктурированным относятся задачи, которые содержат как количественные, так и качественные переменные, причем качественные аспекты проблемы имеют тенденцию доминировать. Неструктурированные проблемы имеют лишь качественное описание.

В СППР дается такое определение: «Система поддержки принятия решений – это компьютерная система, позволяющая ЛПР сочетать собственные субъективные предпочтения с компьютерным анализом ситуации при выработке рекомендаций в процессе принятия решения». Основной пафос этого определения – сочетание субъективных предпочтений ЛПР с компьютерными методами.

В СППР определяется «как компьютерная информационная система, используемая для различных видов деятельности при принятии решений в ситуациях, где невозможно или нежелательно иметь автоматическую систему, полностью выполняющую весь процесс решения».

Системы поддержки принятия решений – это экспертные и автоматизированные системы управления, которые позволяют принимать необходимые решения, сложившиеся в той или иной ситуации.

Системы поддержки принятия решений – это компьютерные системы, предназначенные для сбора и анализа больших объёмов информации с целью оценивания возможных вариантов решений.

Все эти определения не противоречат, а дополняют друг друга и достаточно полно характеризуют СППР.

Человеко-машинная процедура принятия решений с помощью СППР представляет собой циклический процесс взаимодействия человека и компьютера. Цикл состоит из фазы анализа и постановки задачи для компьютера, выполняемой лицом, принимающим решение (ЛПР), и фазы оптимизации (поиска решения и выполнения его характеристик), реализуемой компьютером.

Системы поддержки принятия решений:

1. помогают произвести оценку обстановки (ситуаций), осуществить выбор критериев и оценить их относительную важность;

2. генерируют возможные решения (сценарии действий);

3. осуществляют оценку сценариев (действий, решений) и выбирают лучшую;

4. обеспечивают постоянный обмен информацией об обстановке принимаемых решений и помогают согласовать групповые решения;

5. моделируют принимаемые решения (в тех случаях, когда это возможно);

6. осуществляют динамический компьютерный анализ возможных последствий принимаемых решений;

7. производят сбор данных о результатах реализации принятых решений и осуществляют оценку результатов.

По сути любой механизм, помогающий человеку принимать какие-либо решения, на основе не только интуиции, можно назвать системой поддержки принятия решений. Поэтому элементы систем поддержки принятия решений есть в системе, где есть возможность построить какой-либо график либо получить отчёт. К сожалению, СППР – это всего лишь система и она не может заменить человека, но тем не менее информационная сложность определяется необходимостью учёта большого объёма данных, обработка которых без помощи современных вычислительных машин практически невыполнима. В этих условиях число возможных решений, как правило, весьма велико, и выбор наилучшего из них «на глаз» без всестороннего анализа может приводить к грубым ошибкам.

Основные функции таких систем следующие:

* оценивание проблемной ситуации и имеющихся ограничений;
* выявление приоритетов и их ранжирование;
* формулировка предпочтений лица принимающего решения (ЛПР);
* генерирование альтернатив;
* оценивание альтернатив;
* анализ возможных последствий и рисков;
* выбор наилучшей альтернативы.

Для реализации указанных функций в системах поддержки принятия решений используются следующие методы:

* информационный поиск;
* анализ данных;
* поиск знаний в базах данных;
* имитационное моделирование;
* когнитивное моделирование;
* ситуационное моделирование и др.

Основными компонентами систем поддержки принятия решений являются:

* хранилища данных;
* средства обработки данных;
* многомерные базы данных;
* data mining.

Лицо принимающее решение может использовать системы поддержки принятия решений на всех основных этапах разработки и принятия решений:

* распознавания проблемы;
* генерирования и анализа альтернатив;
* выбор наилучшей альтернативы;
* реализации решений.

Системы поддержки принятия решений помогают решать такие типо-вые задачи:

* анализ примеров;
* идентификация зависимостей;
* анализ чувствительности;
* анализ возможностей;
* анализ влияния;
* анализ данных;
* сравнение и агрегирование;
* анализ рисков;
* оптимизация.

Взаимодействие с ЛПР выделяют три основных класса систем поддержки принятия решений:

* + пассивные;
  + активные;
  + кооперативные.

По способу поддержки различают такие классы систем:

* + модельно-ориентированные СППР. Они используют для своей работы: статистические, экономические и др. типы моделей;
  + коммуникационно-ориентированные СППР. Обеспечивают взаимодействие нескольких пользователей в процессе разработки решения;
  + СППР, ориентированные на данные. Имеют доступ к необходимым базам данных, а также средства для работы с этими базами;
  + СППР, ориентированные на документы. Содержат средства обработки, имеющихся документов деструктурированной информации;
  + СППР, ориентированные на знания. Обеспечивают решение определенных классов задач на основе знаний, которые накоплены в соответствующей предметной области.

По сфере использования выделяют общесистемные и настольные СППР.

Общесистемные СППР ориентированы на взаимодействие с большими системами хранения данных и могут одновременно работать с несколькими, а иногда и со многими пользователями.

Настольные СППР имеют более ограниченные возможности по хранению и обработке информации и предназначены для работы одного пользователя.

С точки зрения архитектуры выделяют-таки типы системы поддержки принятия решений:

* + функциональные СППР. Предназначены для небольших организаций, такие СППР обеспечивают компактность и оперативность работы. Обычно используются для узкого круга проблем, а качество получаемых результатов окажется недостаточным из-за необеспечения предварительной очистки данных;
  + СППР с независимыми витринами данных. Включает несколько витрин данных, каждая из которых предназначена для своего класса задач и ориентирована на определенную группу пользователей, это дает возможность повысить производительность, но приводит к необходимости дублировать данные. Недостаток таких систем, это сложность наполнения витрин и отсутствия консолидации данных, что ведет к отсутствию у ЛПР целостной картины объекта управления;
  + СППР на основе двухуровневых хранилищ данных. Применяются там, где данные консолидированы в единую систему. Это дает возможность предложить унифицированные способы обработки информации, однако, такие СППР не могут обеспечить возможность структурирования данных для отдельных групп пользователей, а также ограничить доступ пользователя для той или иной информации. Кроме того, необходимо создание группы специалистов для обслуживания данной системы;
  + СППР на основе трехуровневых хранилищ данных. Позволяет осуществить консолидацию информации и ее предварительную очистку, а также сформировать витрину данных, которая будет использоваться отдельными группами пользователей. Такие СППР содержат единую корпоративную модель данных и характеризуется гарантированной производительностью, однако данные в таких системах являются избыточными и это ведёт к осложнению хранения этих данных, а также к возникновению проблемы со-голосования архитектуры с различающимися запросами различных групп пользователей.

Общий вид системы поддержки принятия решений выглядит в соответствии с рисунком 1.

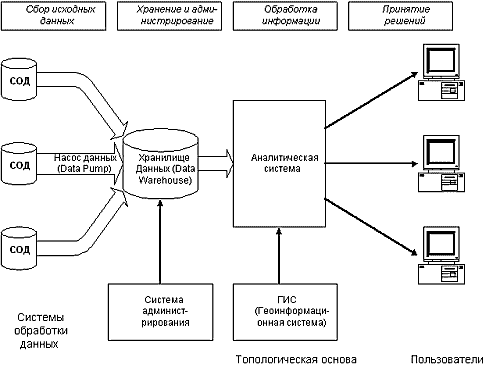


Рисунок 1 – Общий вид СППР

Некоторые примеры задач, решаемые с помощью систем поддержки принятия решений:

* выбор маршрутов перевозок;
* управление пакетами акций;
* управление производственными ресурсами предприятий;
* составление расписаний;
* управление проектами и др.

Системы поддержки принятия решений могли использовать в основном крупные компании, так как разработка подобных систем требует очень огромных ресурсов и времени. С течение некоторого времени эти системы стали доступным и мелким организациям, так как рост информационных технологий не стоит на месте. На сегодняшний день уже абсолютно каждый может использовать и применять подобные системы в любой сфере деятельности для принятия тех или иных решений.

Сейчас системы поддержки принятия решений изменили свой термин на BI-системы. В качестве синонимов систем поддержки принятия решений оперируют понятиями «аналитическая система» и «информационно-аналитическая система» (ИАС). Для обозначения аналитический технологий и средств в целом теперь принято использовать термин "Business Intelligence". Понятие BI объединяет различные средства и технологии анализа и обработки данных масштаба предприятия, на их основе создаются BI-системы. Их цель – повысить качество информации для принятия управленческих решений.

В данный момент класс систем BI является независимым классом систем, в который входят системы класса СППР.

По оценкам International Data Corporation рынок Business Intelligence состоит из 5 сегментов:

* online analytical processing (OLAP)-продукты;
* инструменты добычи данных;
* средства построения Хранилищ и Витрин данных;
* управленческие информационные системы и приложения;
* инструменты конечного пользователя для выполнения запросов и построения отчетов.

Управленческие информационные системы и приложения.

Существует еще один очень разносторонний класс аналитических систем. Это – конечные решения для управленцев и аналитиков. Исторически сложилось так, что технологическая основа реализации таких систем существенно различается. Одни из них построены на современных аналитических инструментах, другие – с применением базовых информационных технологий. Аналитические системы классифицируются по виду задач, решаемых с их помощью. Среди видов задач можно выделить:

* Анализ финансового состояния банка или предприятия. Системы - Audit Expert (Про-Инвест), Альт-финансы (Альт), АБФИ (Вестона), Аналитик, АФСП, АДП (ИНЭК) и другие;
* Инвестиционный анализ – для комплексной оценки эффективности инвестиционных проектов и принятия решения об их финансировании, Project Expert (Про-Инвест), Альт-Инвест (Альт) и другие;
* Подготовка бизнес-планов. Системы - Project Expert (Про-Инвест), Альт-Инвест (Альт) и другие;
* Маркетинговый анализ. Системы - Marketing Expert (Про-Инвест), Касатка и другие;
* Управление проектами. Системы - MS Project (Microsoft), Open Plan (Welcom Software Technology) и другие;
* Бюджетирование. Системы - Hyperion Pillar, Comshare MPC, Контур Корпорация. Бюджет (Intersoft Lab) и другие;
* Финансовое управление. Системы - Oracle Financial Services Applications (Oracle),Контур Корпорация. Финансовое управление (IntersoftLab) и другие.

Инструменты конечного пользователя для выполнения запросов и построения отчетов.

Системы данного класса (Query & Reporting) предназначены для формирования запросов к информационным системам в пользовательских терминах, а также их исполнение, интеграцию данных из разных источников, просмотр данных с возможностями детализации и обобщения и построение полноценных отчетов, как экранных, так и печатных.

Практически каждая система класса OLAP снабжена средствами Query & Reporting. Эти средства могут быть как встроенными в основной продукт (примеры – "Контур Стандарт", Oracle Discoverer), так и выделенными в отдельный продукт (например, система Impromptu).

Также существуют и специализированные системы генерации и дистрибуции отчетов. Наиболее распространенные из них – это продукты компаний Crystal Decisions и Actuate. [10]

В соответствии с рисунком 2, полная структура информационно-аналитической системы, построенной на основе хранилища данных. В конкретных реализациях отдельные компоненты этой схемы часто отсутствуют.



Рисунок 2 – Обработка и анализ информационных потоков

1.2 История развития систем поддержки принятия решений

До середины 60-х годов прошлого века создание больших информационных систем (ИС) было чрезвычайно дорогостоящим, поэтому первые ИС менеджмента (так называемые Management Information Systems – MIS) были созданы в эти годы лишь в достаточно больших компаниях. MIS предназначались для подготовки периодических структурированных отчетов для менеджеров.

В конце 60-х годов появляется новый тип ИС – модель-Ориентированные СППР (Model-oriented Decision Support Systems – DSS) или системы управленческих решений (Management Decision Systems - MDS).

По мнению первооткрывателей СППР Keen P. G. W., Scott Morton M.S. (1978), концепция поддержки решений была развита на основе «теоретических исследований в области принятия решений и технических работ по созданию интерактивных компьютерных систем».

В 1971 г. – опубликована книга Scott Morton‘а, в которой впервые были описаны результаты внедрения СППР, основанной на использовании математических моделей.

1974 г. – в работе дано определение первые ИС менеджмента - MIS: «MIS – это интегрированная человеко-машинная система обеспечения информацией, поддерживающая функции операций, менеджмента и принятия решений в организации. Системы используют компьютерную технику и программное обеспечение, модели управления и принятия решений, а также базу данных».

1975 г. - J.D.C.Little в своей работе предложил критерии проектирования СППР в менеджменте.

1978 г. – опубликован учебник по СППР, в котором исчерпывающе описаны аспекты создания СППР: анализ, проектирование, внедрение, оценка и разработка.

1980 г. – опубликована диссертация S. Alter, в которой он дал основы классификации СППР.

1981 г. – Bonczek, Holsapple и Whinston в книге создали теоретические основы проектирования СППР. Они выделили 4 необходимых компонента, присущих всем СППР: 1) Языковая система (Language System - LS) – СППР может принимать все сообщения; 2) Система презентаций (Presentation System (PS)) (СППР может выдавать свои сообщения); 3) Система знаний (Knowledge System - KS) – все знания СППР сохраняет; 4) Система обработки задач (Problem-Processing System (PPS)) – программный «механизм», который пытается распознать и решить задачу во время работы СППР.

1981 г. – В книге R.Sprague и E.Carlson описали, каким образом на практике можно построить СППР. Тогда же была разработана информационная система руководителя (Executive Information System (EIS)) – компьютерная система, предназначенная для обеспечения текущей адекватной информации для поддержки принятия управленческих решений менеджером.

Начиная с 1990-х, разрабатываются так называемые Data Warehouses - хранилища данных. Хранилище данных – это очень большая предметно- ориентированная информационная корпоративная база данных, предназначенная для подготовки отчётов, анализа бизнес-процессов и поддержки принятия решений. Строится на базе клиент-серверной архитектуры, реляционной СУБД и утилит поддержки принятия решений. Данные, поступающие в хранилище данных, становятся доступны только для чтения.

27 октября 2005 года в Москве на Международной конференции «Информационные и телемедицинские технологии в охране здоровья» (ITTHC 2005), А. Пастухов (Россия) представил СППР нового класса - PSTM (Personal Information Systems of Top Managers). Основным отличием PSTM от существующих СППР является построение системы для конкретного лица, принимающее решение, с предварительной логико-аналитической обработкой информации в автоматическом режиме и выводом информации на один экран.

Интересно отметить создание предтечи СППР коллежским советником С. Н. Корсаковым, опубликовавшим еще в 1832 году описание механических устройств, так называемых «интеллектуальных машин», которые "могли быть использованы при решении различных задач в повседневной жизни, для того, чтобы сделать какой бы то ни было вывод", например, помочь принять решение о наиболее подходящих лекарствах по наблюдаемым у пациента симптомам заболевания.

1.3 Метод анализа иерархий

Метод анализа иерархий (МАИ) является структурированным методом для организации и анализа сложных решений, основанных на математике и психологии. Он был разработан Томасом Л. Саати в 1970-х годах и с тех пор изучен, и уточнен.

Он имеет особое применение в принятии решений по группам [1] и используется во всем мире в самых разных ситуациях принятия решений в таких областях, как правительство, бизнес, промышленность, здравоохранение, судостроение и образование.

Вместо того, чтобы предписывать «правильное» решение, МАИ помогает лицам, принимающим решения, найти то, которое наилучшим образом соответствует их цели и пониманию проблемы. Этот метод обеспечивает всеобъемлющую и рациональную структуру для структурирования проблемы решения, для представления и количественной оценки ее элементов, для связывания этих элементов с общими целями и для оценки альтернативных решений.

Пользователи МАИ сначала разлагают свою проблему решения в иерархию более понятных подзадач, каждая из которых может анализироваться независимо. Элементы иерархии могут относиться к любому аспекту проблемы решения - материальному или неосязаемому, тщательно измеренному или грубо оцениваемому, хорошо или плохо понимаемому - что-либо вообще применимое к решению.

Как только иерархия построена, руководители системы систематически оценивают ее различные элементы, сравнивая их друг с другом по два за раз, в зависимости от их влияния на элемент, находящийся над ними в иерархии. При проведении сравнений лица, принимающие решения (ЛПР), могут использовать конкретные данные об элементах, но обычно используют свои суждения об относительном значении и значимости элементов. Суть МАИ заключается в том, что человеческие суждения, а не только основная информация, могут использоваться при проведении оценок.

МАИ преобразует эти оценки в числовые значения, которые можно обрабатывать и сравнивать по всему диапазону проблемы. Численный вес или приоритет выведен для каждого элемента иерархии, позволяя рационально и последовательно сравнивать разнообразные и часто несоизмеримые элементы друг с другом. Эта возможность отличает МАИ от других методов принятия решений.

На последнем этапе процесса вычисляются числовые приоритеты для каждой из альтернатив решения. Эти цифры представляют собой относительную способность альтернатив к достижению цели решения, поэтому они позволяют прямо рассматривать различные направления действий.

Несколько фирм поставляют компьютерное программное обеспечение, чтобы помочь в использовании этого процесса.

1.4 Использование и применение МАИ

Хотя это метод может использоваться людьми, которые работают над прямыми решениями, метод анализа иерархии наиболее полезен, когда команды людей работают над сложными проблемами, особенно с высокими ставками, с участием человеческих представлений и суждений, решения которых имеют долгосрочные последствия. Он имеет уникальные преимущества, когда важные элементы решения трудно количественно оценить или сравнить, или когда общение между членами команды затруднено их различными специализациями, терминологией или перспективами.

Решения, к которым может быть применено МАИ, включают:

* Выбор - выбор одной альтернативы из заданного набора альтернатив, обычно при наличии нескольких критериев принятия решения;
* Ранжирование. Помещение набора альтернатив в порядке от большинства до наименее желательных;
* Приоритезация. Определение относительного достоинства членов набора альтернатив, в отличие от выбора одного или просто их ранжирования;
* Распределение ресурсов - распределение ресурсов между набором альтернатив;
* Бенчмаркинг - сравнение процессов в собственной организации с другими лучшими в своем роде организациями;
* Управление качеством - работа с многомерными аспектами повышения качества и улучшения этих качеств;
* Разрешение конфликтов - урегулирование споров между сторонами с явно несовместимыми целями или позициями.

Применение МАИ к сложным ситуациям принятия решений пронумерованы в тысячах изданий и дали обширные результаты в проблемах, связанных с планированием, распределением ресурсов, установкой приоритетов и выбором среди альтернатив. Другие области включают прогнозирование, общее управление качеством, реинжиниринг бизнес-процессов, развертывание функций качества и сбалансированную систему показателей. Многие приложения МАИ никогда не сообщаются в мире в целом, поскольку они происходят на высоких уровнях крупных организаций, где соображения безопасности и конфиденциальности запрещают их раскрытие. Но некоторые виды использования МАИ обсуждаются в литературе. В последнее время они включали:

* Выбор типа ядерных реаторов (Политехнический университет Милано);[1]
* Решение о том, как наилучшим образом снизить воздействие глобального изменения климата (Fondazione Eni Enrico Mattei); [2]
* Количественное определение общего качества программных систем (Microsoft Corporation); [3]
* Выбор университетского факультета (Блумсбергский университет Пенсильвании); [4]
* Оценка риска при эксплуатации нефтепроводов по пересеченной местности (Американское общество инженеров-строителей); [5]
* Решая, как лучше всего управлять водоразделами США (Министерство сельского хозяйства США). [6]

МАИ иногда используется при разработке высокоспециализированных процедур для конкретных ситуаций, таких как рейтинг зданий по историческому значению. Он был применен к проекту, который использует видеоматериалы для оценки состояния автомобильных дорог в Вирджинии. Инженеры-шоссейники сначала использовали его для определения оптимального объема проекта, а затем для оправдания его бюджета законодателям. [7]

В России министерство здравоохранения и социального Развития российской федерации Научный центр психического здоровья РАМН применил и описал методо МАИ в программе «применение метода анализа иерархий в практике психосоциальной реабилитации и в программе дестигматизации». [8]

1.5 Образование и научные исследования МАИ

Хотя использование процесса аналитической иерархии не требует специализированной академической подготовки, оно считается важным предметом во многих высших учебных заведениях, включая инженерные школы и аспирантуру бизнеса. Это особенно важный предмет в области качества и преподается на многих специализированных курсах.

Значимость МАИ признана в развитых и развивающихся странах по всему миру. Пример Китая - около сотни китайских университетов предлагают курсы в МАИ, и многие докторанты выбирают МАИ как предмет своих исследований и диссертаций. В Китае опубликовано более 900 работ по этому вопросу, и нет ни одного китайского научного журнал, посвященного исключительно этому методу.

Международный симпозиум по аналитической иерархии (ISAHP) проводит двухгодичные встречи ученых и практиков, заинтересованных в этой области. Раскрывается широкий круг тем. Те, кто в 2005 году, варьировались от установления стандартов оплаты для хирургических специалистов, разработки стратегических технологий, реконструкции инфраструктуры в опустошенных странах. На встрече 2007 года в Вальпараисо, Чили, более 90 докладов были представлены из 19 стран, включая США, Германию, Японию, Чили, Малайзию и Непал. Аналогичное количество докладов было представлено на симпозиуме 2009 года в Питтсбурге, штат Пенсильвания, когда было представлено 28 стран. Тематика статей включала экономическую стабилизацию в Латвии, подбор портфеля в банковском секторе, управление лесными пожарами, чтобы помочь смягчить глобальное потепление и сельские микропроекты в Непале.

1.6 Пример задачи многокритериального выбора с простейшей иерархией МАИ

В данной задаче необходимо выбрать из трех кандидатов одного на должность руководителя. Кандидаты оцениваются по критериям: возраст, опыт, образование и личные качества. В соответствии с рисунком 3 показана иерархия для этой задачи. Простейшая иерархия содержит три уровня: цель, критерии и альтернативы. Числа на рисунке показывают приоритеты элементов иерархии с точки зрения цели, которые вычисляются в МАИ на основе парных сравнений элементов каждого уровня относительно связанных с ними элементами вышерасположенного уровня. Приоритеты альтернатив относительно цели (глобальные приоритеты) вычисляются на заключительном этапе метода путём линейной свертки локальных приоритетов всех элементов. В данном примере лучшим кандидатом является Дик, так как имеет максимальное значение глобального приоритета.

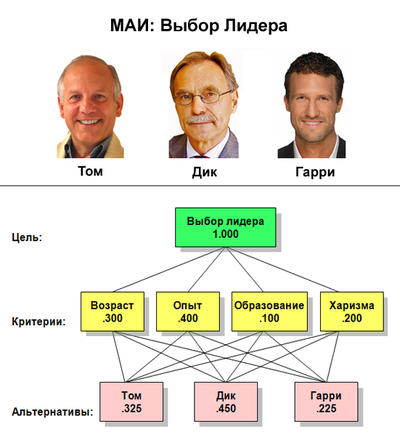


Рисунок 3 – МАИ. Выбор лидера

1.7 Сфера образования и научных исследований МАИ

Хотя для практического применения МАИ отсутствует необходимость специальной подготовки, основы метода преподают во многих учебных заведениях. Кроме того, этот метод широко применяется в сфере управления качеством и читается в рамках многих специализированных программ, таких как Six Sigma, Lean Six Sigma, и QFD. Около ста китайских университетов предлагают курсы по основам МАИ, и многие соискатели научных степеней выбирают МАИ в качестве объекта научных и диссертационных исследований. Опубликовано более 900 научных статей по данной тематике. Существует китайский научный журнал, специализирующийся в области МАИ. Раз в два года проводится Международный симпозиум, посвященный МАИ (International Symposium on Analytic Hierarchy Process, ISAHP), на котором встречаются как ученые, так и практики, работающие с МАИ. В 2007 году симпозиум проходил в Вальпараисо, Чили, где было представлено более 90 докладов ученых из 19 стран, включая США, Германию, Японию, Чили, Малайзию, и Непал.

1.8 Методика применения МАИ

Метод анализа иерархий содержит процедуру синтеза приоритетов, вычисляемых на основе субъективных суждений экспертов. Число суждений может измеряться дюжинами или даже сотнями. Математические вычисления для задач небольшой размерности можно выполнить вручную или с помощью калькулятора, однако гораздо удобнее использовать программное обеспечение (ПО) для ввода и обработки суждений. Самый простой способ компьютерной поддержки — электронные таблицы, самое развитое ПО предусматривает применение специальных устройств для ввода суждений участниками процесса коллективного выбора. Порядок применения Метода Анализа Иерархий:

1. Построение качественной модели проблемы в виде иерархии, включающей цель, альтернативные варианты достижения цели и критерии для оценки качества альтернатив.
2. Определение приоритетов всех элементов иерархии с использованием метода парных сравнений.
3. Синтез глобальных приоритетов альтернатив путём линейной свертки приоритетов элементов на иерархии.
4. Проверка суждений на согласованность.
5. Принятие решения на основе полученных результатов.

Моделирование проблемы в виде иерархии. Первый шаг МАИ — построение иерархической структуры, объединяющей цель выбора, критерии, альтернативы и другие факторы, влияющие на выбор решения. Построение такой структуры помогает проанализировать все аспекты проблемы и глубже вникнуть в суть задачи.

1.9 Рынок СППР

На рынке СППР компании предлагают следующие виды услуг по созданию систем поддержки принятия решений:

* Реализация пилот-проектов по СППР-системам, с целью демонстрации руководству Заказчика качественного потенциала аналитических приложений;
* Создание совместно с Заказчиком полнофункциональных СППР-систем, включая хранилище данных и средства Business Intelligence;
* Проектирование архитектуры хранилища данных, включая структуры хранения и процессы управления;
* Создание «витрин данных» для выделенной предметной области;
* Установка и настройка средств OLAP и Business Intelligence; их адаптация к требованиям Заказчика;
* Анализ инструментов статистического анализа и «добычи данных» для выбора программных продуктов под архитектуру и потребности Заказчика;
* Интеграция систем СППР в корпоративные интранет-сети Заказчика, автоматизация электронного обмена аналитическими документами между пользователями хранилища;
* Разработка Информационных Систем Руководителя (EIS) под требуемую функциональность;
* Услуги по интеграции баз данных в единую среду хранения информации;
* Обучение специалистов Заказчика технологиям хранилищ данных и аналитических систем, а также работе с необходимыми программными продуктами;
* Оказание консалтинговых услуг Заказчику на всех стадиях проектирования и эксплуатации хранилищ данных и аналитических систем;
* Комплексные проекты создания/модернизации вычислительной инфраструктуры, обеспечивающей функционирование СППР: решения любого масштаба, от локальных систем до систем масштаба предприятия/концерна/отрасли.

1.10 Состояние Российского рынка Business Intelligence

По оценкам TAdviser, [11] объем российского рынка BI-систем в 2015 году, включая услуги по внедрению и поддержке, увеличился на 5 процентов и достиг 45,9 млрд рублей. В соответствии с рисунком 4 отображена информация динамики рынка BI в России.

Рисунок 4 – Динамика Российского рынка BI, млрд руб.

Рынок продолжает расти вслед за ростом количества различных информационных систем, которые используют заказчики, и объема данных, обрабатываемых в этих системах. Постепенное накопление данных является драйвером спроса на BI-инструменты.

Евгений Смирнов, заместитель генерального директора Navicon, замечает, что с ростом объема данных, в том числе неструктурированных, появляется интерес к математическим и алгоритмическим методам их обработки. В тоже время стоимость новых технологий, датчиков и ИТ-мощностей стремительно снижается во всем мире, появляется возможность использования облачных технологий для обработки данных. «Аналитика становится одним из факторов конкурентоспособности». [11]

Что касается динамики рынка, то большинство его участников говорит о движении вверх на уровне 5-10 процентов.

К примеру, Юрий Востриков, заместитель коммерческого директора «Норбит», полагает, что последние годы российский рынок BI-решений показывает стабильный рост в пределах 10 процентов год к году, если смотреть на него с точки зрения рублевой выручки игроков.

Роман Баранов из «Крок», также считает, что несмотря на негативное влияние экономических процессов, рынок BI растет в денежном выражении.

По мнению Павла Адылина, исполнительного директора Artezio, темпы прироста российского рынка BI колеблется в пределах 5 процентов в рублевом выражении, что связано с сокращением ИТ-бюджетов. При этом до начала экономического кризиса прирост, по его данным, достигал 20 процентов.

2 Публичные API

2.1 API один из новых каналов взаимодействия

Для того чтобы реализовать систему поддержки принятия решений, необходимо использовать публичные API. Такой метод с применением API позволяет получать доступ к различным сервисам и брать информацию для принятия решений из публичного доступа, что значительно облегчает работу при программной реализации человеко-машинного интерфейса.

API (Application Programming Interface) — это интерфейс программирования, интерфейс создания приложений.

Из определения API – это в первую очередь интерфейс, который позволяет разработчикам использовать готовые блоки для создания приложений. Например, с разработкой мобильных приложений в роли API может выступать библиотека для работы с «умным домом», всё практически реализовано в этой библиотеке, а программист лишь обращается к этому API в своем коде.

API представляет собой REST приложение, принимающее и возвращающее данные в формате JSON. С его помощью можно получить список событий, категорий, организаций и связанной с ними информации.

Есть публичные API и большинство таких на западном сегменте, которые предоставляют доступ к своим базам. На просторах рунета ощущается их явный недостаток, однако имеется некоторый список таких сервисов.

Популярные публичные API в сегменте РФ:

* yandex.ru;
* vk.com;
* mail.ru;
* ok.ru.

В мире:

* google.com;
* twitter.com;
* facebook.com;
* sandspace.com;
* openweathemap.com;
* github.com.

Публичная информация отдается через API без необходимости авторизации. Для получения доступа к непубличной информации, либо для изменения данных, нужна авторизация по протоколу OAuth2.

Стоит отметить, что не все публичные API дают полный доступ к своим сервисам, например, сервис «OpenWeatheMap.com» из которого берётся информация о погоде в реализованной среде человеко-машинного взаимодействия, позволяет делать 60 запросов в минуту. В соответствии с рисунком 5, выделен пункт о предоставлении данной услуги.

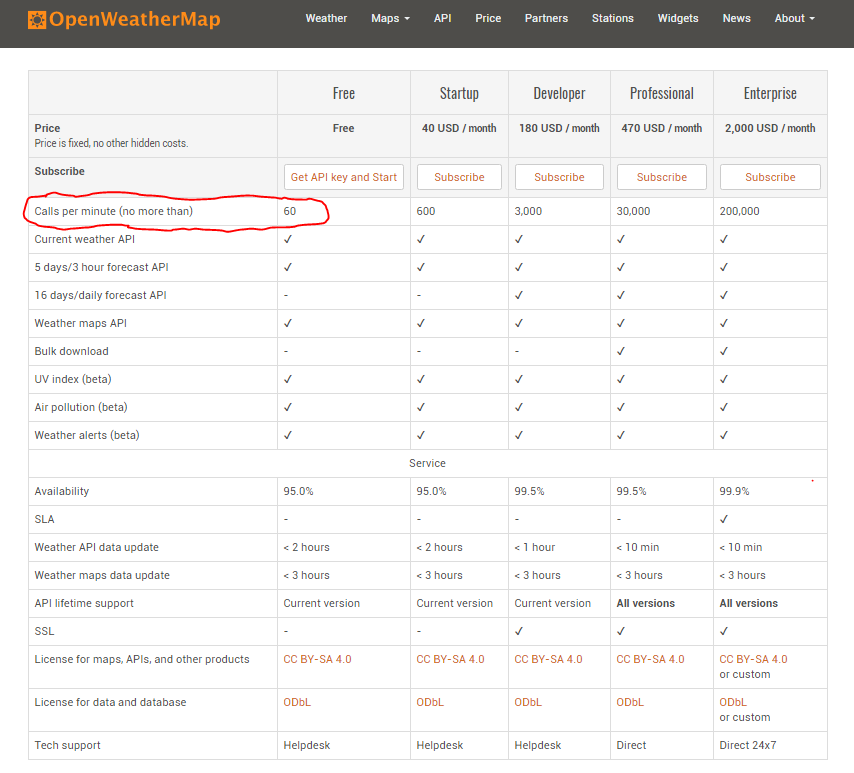


Рисунок 5 – Предоставление 60 запросов в минуту для публичного API

Просмотрев остальные сервисы, стоит отметить, что у всех стоит ограничение по запросам. Сервис «yandex.ru» предоставляет доступ к своим API и количество запросов в сутки 25000 в соответствии с рисунком 6, однако, ограничено некоторое количество функций. Так же предоставлены тарифы по использованию данного сервиса по предоставлению полного списка услуг. В соответствии с рисунком 7 предоставлена ценовая категории, также можно посчитать стоимость того, сколько запросов необходимо использовать. Данная ценовая категория относится к «яндекс картам», но не сильно отличается от других сервисов данного ресурса.

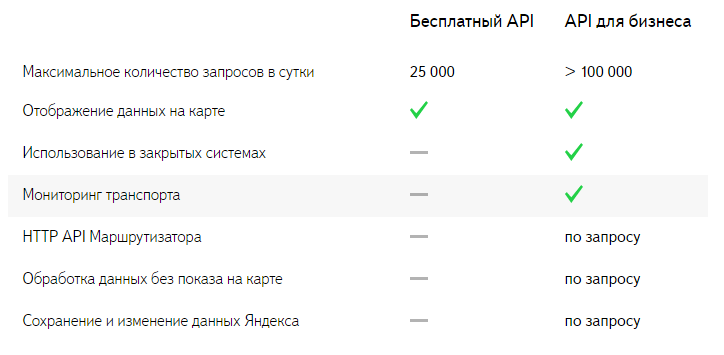


Рисунок 6 – Предоставление бесплатного API c сервиса yandex.ru

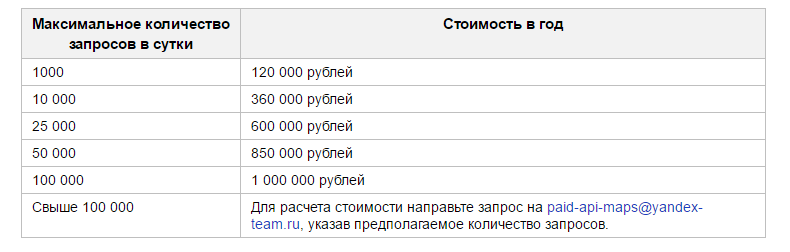


Рисунок 7 – Стоимость услуг сервиса «яндекс карты»

2.2 История API

«Концепция API предваряет даже появление персональных компьютеров, не говоря уже о Web, очень долгое время! Принцип хорошо документированного набора общедоступных «точек входа», которые позволяют приложению взаимодействовать с другой системой является неотъемлемой частью разработки программного обеспечения с самых ранних дней обработки служебных данных. Однако появление распределенных систем, а затем самой сети показало, что важность и полезность этих же базовых концепций резко возрастает». - Мартин Бартлетт

В 2017 году, есть некоторые очевидные технические причины того, почему веб-API находят успех в компаниях всех форм и размеров и даже внутри правительства, но не все причины этого успеха являются техническими. Есть много других, менее очевидных аспектов веб-APis, которые способствовали их успеху, чему можно только учиться, внимательно изучая прошлое и глядя на то, почему некоторые из первопроходцев веб-API были успешными, и продолжают добиваться успеха из года в год.

2.3 Применение публичных API

Для наглядности работы API предложено рассмотреть пример работы OpenWeatherMap API.

OpenWeatherMap API возвращает информацию о текущей метеорологической обстановке пользователю. Данные доступны в формате JSON, XML или HTML.

Запрос можно сделать по названию города или по названию и коду города. API выведет информацию согласно введенным данным.

Запросы API:

*api.openweathermap.org/data/2.5/weather?q={city name},*

*api.openweathermap.org/data/2.5/weather?q={city name},{country code}.*

Параметры:

*q*, *city name* и *country code* разделены запятой, использовать коды стран согласно стандарту ISO 3166.

Проверка запроса:

[*http://api.openweathermap.org/data/2.5/weather?q=Krasnoyarsk&units=metric&appid=acfbe31a83338fceb055e362de64c627*](http://api.openweathermap.org/data/2.5/weather?q=Krasnoyarsk&units=metric&appid=acfbe31a83338fceb055e362de64c627),

где *q* отвечает за название города, в данном случае это город Красноярск, параметр units отвечает за отображение температуры в удобном формате, если это Фаренгейты *units=imperial*, если Градусы *units=metric*, если нужно отображение в единице измерения Кельвин, тогда параметр *units* можно опустить в вызове команды API. Параметр *appid* позволяет использовать полученный ключ для работы с запросами API, без данного ключа *acfbe31a83338fceb055e362de64c627* не получиться выполнить запрос, такова политика публичного данного ресурса и остальных других. Так как это позволяет дать доступ к получению данных с ресурса OpenWeatherMap.

Ответ на запрос в формате JSON:

*{"coord":{"lon":92.79,"lat":56.01},*

*"weather":[{"id":520,"main":"Rain","description":"light intensity shower rain","icon":"09d"}],*

*"base":"stations",*

*"main":{"temp":15,"pressure":1013,"humidity":77,"temp\_min":15,"temp\_max":15},*

*"visibility":10000,*

*"wind":{"speed":5,"deg":230},*

*"clouds":{"all":40},*

*"dt":1497243600,*

*"sys":{"type":1,"id":7285,"message":0.0138,"country":"RU","sunrise":1497214962,"sunset":1497278111},"id":1502026,"name":"Krasnoyarsk","cod":200}*

Для просмотра температуры необходимо просмотреть параметр *main.temp*, который имеет значение 15, что означает в городе Красноярск 15 градусов тепла. В соответствии с рисунком 8 видно, что в Красноярске, как и в запросе отображена верная температура.

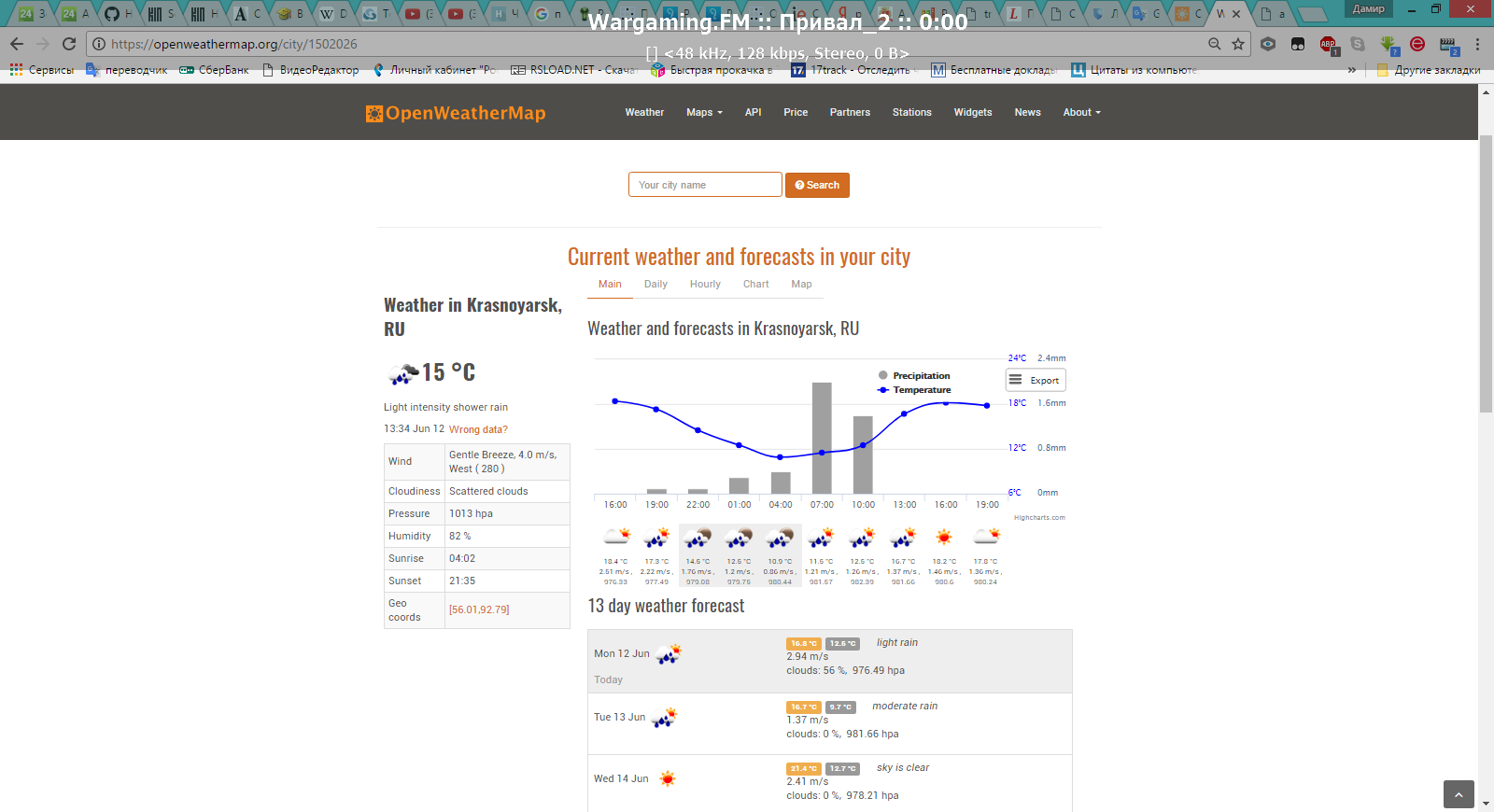


Рисунок 8 – Температура в городе Красноярск

Более подробно о параметрах в текущем запросе описано ниже.

Параметры запроса JSON:

* coord;
* coord.lon. Местоположение города, долгота;
* coord.lat. Географическое местоположение города, широта;
* weather. (дополнительные коды погоды);
* weather.id. Идентификатор условия погоды
* weather.main. Группа параметров погоды (дождь, снег, экстрим и т.д.);
* weather.description. Погодные условия в группе;
* weather.icon. Идентификатор значка погоды;
* base. Внутренний параметр;
* main;
* main.temp. Температура. Единица измерения по умолчанию: Кельвин, metric: Цельсий, imperial: Фаренгейт;
* main.pressure. Атмосферное давление (на уровне моря, если нет данных sea\_level или grnd\_level), гПа;
* main.humidity. Влажность, проценты;
* main.temp\_min Минимальная температура на данный момент. Это отклонение от текущей температуры, которая возможна для крупных городов и мегаполисов. Единица измерения по умолчанию: Кельвин, metric: Цельсий, imperial: Фаренгейт;
* main.temp\_max. Максимальная температура на данный момент. Это отклонение от текущей температуры, которая возможна для крупных городов и мегаполисов. Единица измерения по умолчанию: Кельвин, metric: Цельсий, imperial: Фаренгейт;
* main.sea\_level. Атмосферное давление на уровень моря, гПа;
* main.grnd\_level. Атмосферное давление на уровне земли, гПа;
* wind;
* wind.speed. Скорость ветра. Единица измерения по умолчанию: метр / с, metric: метр / сек, imperial: мили / час;
* wind.deg. Направление ветра, градусы (метеорологическое);
* clouds;
* clouds.all. Облачность, процентов;
* rain;
* rain.3h. Объем дождя за последние 3 часа;
* snow;
* snow.3h. Объем снега за последние 3 часа;
* dt. Время вычисления данных, unix, UTC;
* sys;
* sys.type. Внутренний параметр;
* sys.id. Внутренний параметр;
* sys.message. Внутренний параметр;
* sys.country. Код страны (GB, JP и т.д.);
* sys.sunrise. Время восхода, unix, UTC;
* sys.sunset. Время захода солнца, unix, UTC;
* id. Идентификатор города;
* name. Название города;
* cod. Внутренний параметр.

Данный ресурс отображает погоду в интерфейсе человеко-машинного взаимодействия в виде бота для мессенджера telegram. А также используется ещё один ресурс для получения данных метеорологической службы это weather.com, на случай если текущий ресурс будет не доступен.

3 Рекомендационные сервисы

3.1 Определения и применение рекомендационных сервисов

Рекомендационные сервисы – это ещё один вид систем поддержки принятия решений, который помогает пользователям с некоторым выбором по заданным критериям.

Рекомендационная система – это программный комплекс, который определяет интересы и предпочтения посетителя и дает рекомендации в соответствии с ними.

Интерес к рекомендательным сервисам возник после соревнования Netflix Prize, организованного компанией Netflix. В конце 90-х годов Netflix рассылала по почте VHS-кассеты. Позднее она создала рекомендательный интернет-сервис Video on Demand. Чем лучше сервис рекомендовал пользователям фильмы, тем больше кассет и дисков они брали в прокат. Со временем компании потребовалось повысить качество рекомендаций, и именно с этой целью в 2006 году придумали Netflix Prize. Netflix выложила в открытый доступ собранные данные – около 100 миллионов оценок по пятибалльной шкале с указанием ID пользователей, проставивших их, а всем желающим предлагалось улучшить результат. Соревнование длилось почти три года. Победители получили приз в $ 1 млн.

Зачем вообще нужно данное направление сервисов? На различных этапах развития Интернета, как глобального информационного пространства, все чаще и чаще поднимаются вопросы о создании качественных сайтов посредников, помогающих потребителям информации взаимодействовать с ней. А именно: облегчать жизнь при поиске данных, их разбивки на категории, сортировки, хранении и т.д. Многие подобные задачи давно успешно решены, а некоторые только начинают решаться.

В Интернете существует огромное количество поисковых систем, каталогов, рейтингов, файлообменников. В результате имеем постоянно увеличивающееся количество контента, ведь в основном каждый сайт-сервис представлен определенной порцией информации, порождающей в процессе своего функционирования еще большие её объемы. Разорвать данный замкнутый круг и помочь потребителю контента сориентироваться в его разнообразии, призваны как раз рекомендательные сервисы. Это является их основной, но отнюдь не единственной задачей.

Дело в том, что большое количество всевозможной информации окружает человека не только в Интернете, а еще и в реальной жизни. Каждый индивидуум по-разному её воспринимает и отфильтровывает, согласно своим личным вкусам и предпочтениям. И действительно, если бы люди пытались осмыслить, распознать, употребить все то, что их окружает в мире, им бы банально не хватило на это времени. Тем не менее, любой объект Вселенной в каждом конкретном случае можно однозначно представить в виде определенных информационных характеристик и затем перенести полученные таким образом данные на удобный носитель. Постепенно таким универсальным носителем становится Интернет.

Максимально упорядочить, выделить и сохранить лишь основные указатели на определенные порции данных, интересные и полезные отдельно взятому пользователю – вот основной общий набор задач, которые должны решать рекомендательные сервисы. Как и поисковые машины, рекомендательные сервисы должны являться для пользователя одновременно связующим звеном и ограждающим фильтром. Можно проводить множество аналогий рекомендательных сервисов с другими существующими средствами взаимодействия с информацией, однако обычно они рассматриваются лишь как малая составляющая ресурсов, специализирующихся на рекомендациях. Если еще проводить аналогию с существующими сервисами, то рекомендующие сайты – это своего рода “суперзакладки”, содержащие ссылки на самые различные массивы данных. В идеале, рекомендательный сервис–это инструмент, позволяющий максимально эффективно оперировать информацией об информации.

Стоит упомянуть, что рекомендательные ресурсы предоставляют как положительные, так и отрицательные оценки по отдельности либо одновременно.

Нельзя сказать, что рекомендательные сервисы будут панацеей от всех информационных "бед" пользователей цифровых ресурсов, хотя на данный момент они как никогда близки к этому. Основные недостатки таких систем – рекомендующие сайты, как и многие другие ресурсы, где контент создается и “фильтруется” пользователями, подвержены спаму (хотя и в меньшей степени, чем в других сервисах) и неадекватным оценкам и рекомендациям (человеческий фактор). Возможно, когда-нибудь в будущем, одной из ступени развития вебтехнологий станут рекомендательные сервисы, основанные на работе искусственного интеллекта. Тогда конечно задача борьбы со спамом будет близка к полному решению.

3.2 Список рекомендательных сервисов

В Российском сегменте первым рекомендательным сервисом стал ИМХОНЕТ, основанный в 2007 году. Сервис одновременно работал с разными видами контента. Создатели назвали его мультикультурным, потому что он помогал пользователю сделать выбор в разных культурных и околокультурных областях, рекомендуя книги, фильмы, игры и прочее.

На сегодняшний день лучшие рекомендательные системы реализованы у нескольких компаний:

Amazon.com – интернет-магазин, покупатели которого могут оставлять отзывы о товарах и оценивать их, что, в свою очередь, в дальнейшем помогает выбрать нужную вещь хорошего качества. В соответчики с рисунком 9 выбор категорий на выбор;

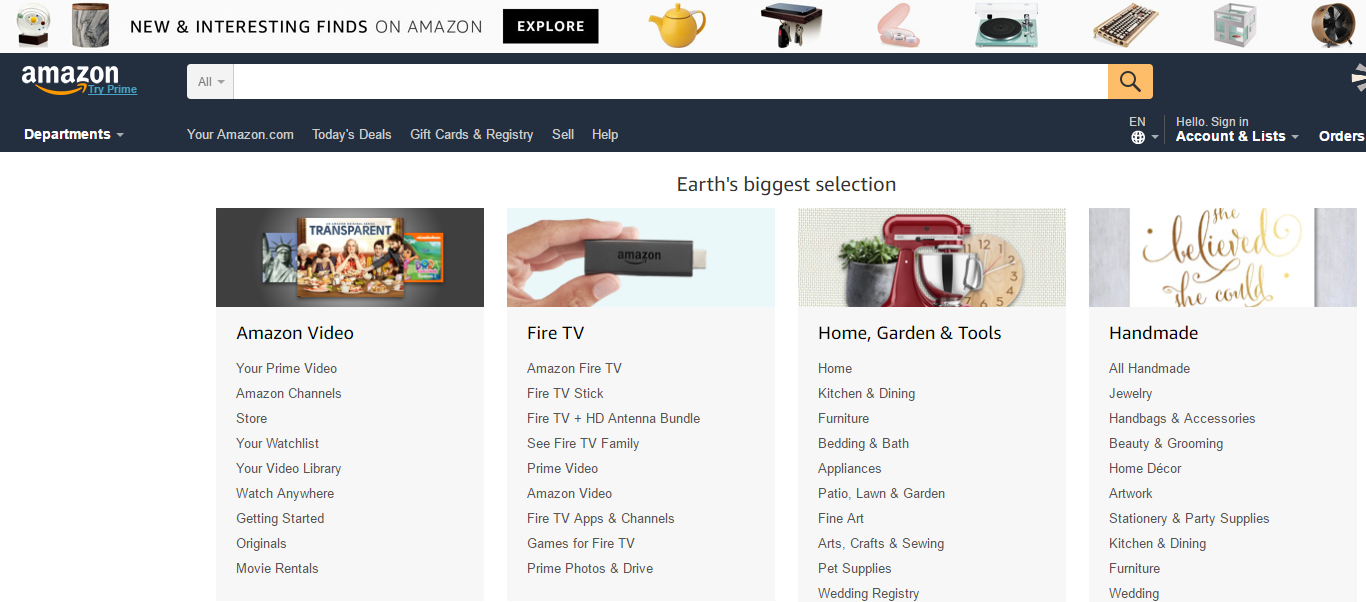


Рисунок 9 – Подбор товаров по категориям

Rich Relevance – рекомендательная платформа для мультиканальной персонализации интернет-магазинов, созданная разработчиками Amazon.com;

Retail Rocket – российская рекомендательная платформа для мультиканальной персонализации интернет-магазина на основе big data, созданная разработчиками рекомендательных систем http://ozon.ru и http://wikimart.ru, аналог Rich Relevance;

Ozon.ru — онлайн-мегамаркет, один из первых российских проектов электронной коммерции в сегменте b2c. Открыт в 1998 году. Ассортимент — 3,5 млн наименований товаров. Ежедневная посещаемость — 700 тыс. пользователей. Штат — около 2000 сотрудников. Оборот — $750 млн (в 2013 году). Официальный сайт — [www.ozon.ru](http://www.ozon.ru). В соответствии с рисунком 10 так же отображена большая база по категориям выбора товара, крупный рекомендательный сервис на российском сегменте.

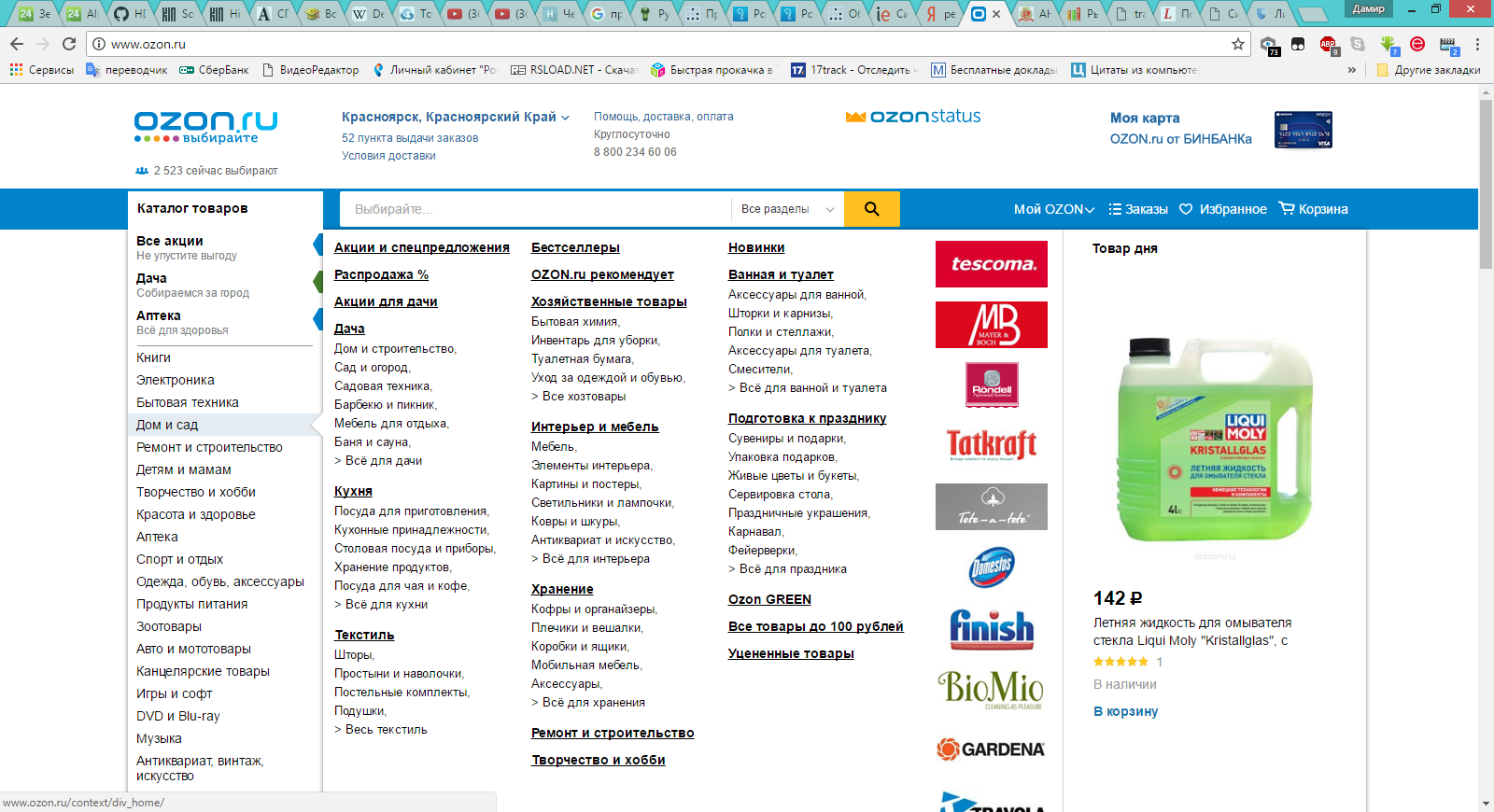


Рисунок 10 – Выбор товаров на Ozon.ru

Любое использование материалов допускается только при наличии гиперссылки.

Так же стоит отметить следующие рекомендательные сервисы, которые имеют значительную популярность:

Ebay – рекомендательная платформа предоставляющая услуги в областях интернет-аукционов (основное поле деятельности) и интернет-магазинов.

Компания eBay представила в России "Коллекции" — новый рекомендательный сервис для покупателей. "Коллекции" на eBay дают покупателям возможность создавать персонализированную подборку товаров, объединенных одной темой или идеей. Посетители сайта могут формировать коллекции из одежды, предметов интерьера, украшений или аксессуаров, и выкладывать свои подборки на странице eBay, предлагая тем, кто ищет нечто подобное — свои "находки";

Mail.ru Group запустил новое приложение Likemore – виртуальный проводник в мире новостей. В основу Likemore лег принцип новостного агрегатора. Пользователь самостоятельно выбирает интересующие его категории из предложенного списка и тут же получает актуальные публикации с отобранных информационных ресурсов;

Yandex.ru – является рекомендательным сервисом с несколькими направлениями, такими как Диско, Яндекс.Радио, Дзен, Маркет;

Lastfm – сервис музыкальных рекомендаций, предоставляющий персональные рекомендации в соответствии с твоими музыкальными предпочтениями. С помощью Last.fm можно найти для себя какие-то новые музыкальные композиции;

Имхонет – Популярный и сравнительно давно работающий рекомендательный сервис Рунета. Имеет множество разделов касающихся рекомендаций самых различных направлений. Сервис обладает привлекательным интерфейсом со множеством анимированных блоков. Однако на сегодняшний день данный ресурс более не доступен, на главной странице сайта присутствует следующее сообщение: «Пионер рекомендательных сервисов в России. Теперь, когда они есть везде, могу уйти в длительный отпуск. Люблю всех! Пока!»;

Brandz – Cравнительно новый ресурс, в основе которого список вещей и брендов, которые пользователи могут оценивать, продавать, менять, дарить и даже желать. На главной странице проекта находятся рейтинги популярности брендов;

Brandz позволяет отследить появление нового товара в продаже и его потребительские характеристики. Понятное дело, что недостатки большинства товаров выявляются после их приобретения, но воспользовавшись подобным сервисом можно застраховаться от множества чужих ошибок;

Рецензент – сайт, содержащий рецензии и авторские оценки кинофильмов, книг и музыкальных произведений. Имеет довольно простой, но быстрый дизайн. Копирайты и счетчики на главной странице отсутствуют;

Обзограф – данный сервис будет чрезвычайно полезен потребителям, которые планируют всевозможные покупки, походы в развлекательные заведения, турпоездки по разным странам или тем, кто не определился с выбором и готов воспользоваться советами продвинутых пользователей.

Feedme — плагин для браузера Firefox, который тоже по праву может считаться сервисом рекомендаций. Он сохраняет информацию, на каких сайтах вы проводили больше всего времени, а затем предоставляет свои рекомендации;

Imdb – это база профайлов кинофильмов и оценок к ним. Самая большая база по этой тематике. Рейтинг на imdb решающий фактор для многих при выборе фильма для просмотра.

Если говорить об эволюции уже существующих рекомендательных сервисов, то можно смело заявить, что их основная масса занимается улучшением своих технологий и алгоритмов и лишь немногие рекомендательные сервисы превратились в платформу.

Новые рекомендательные сервисы – это в основном приложения, тесно связанные с офлайном. Это когда действие совершается онлайн, дабы получить товар или услугу в реальной жизни. Например, бронь билета на предложенный спектакль, чтобы сходить в театр, или выбор ресторана и так далее.

В последнее время рекомендательных сервисов появилось довольно много, но многие ниши все еще не заняты, и пока есть что развивать и чем заниматься.

Рекомендательные сервисы являются одним из самых перспективных направлений. Которые облегчают запросы пользователя по заданным критериям.

4 Выбор системы управления базами данных

4.1 История PostgreSQL

Немаловажно выбрать нужную СУБД для реализации бота мессенджера, необходимо иметь такую БД, которая будет работать с большим количество подключений, поэтому подходящий вариант это PostgreSQL.

PostgreSQL, первоначально названный Postgres, был создан в UCB профессором компьютерной науки по имени Майкл Стоунбрейкер, который стал техническим директором Informix Corporation. Стоунбрейкер начал разрабатывать Postgres в 1986 году в качестве последующего проекта своему предшественнику, Ingres, которому теперь принадлежит Computer Associates. Название Postgres таким образом отрывается от своего предшественника (как в «после Ingres»). Ingres, разработанный с 1977 по 1985 год, был упражнением в создании системы баз данных в соответствии с классической теорией РСУБД. Postgres, разработанный в период между 1986-1994 годами, был проектом, призванным открыть новые основы в концепциях базы данных, таких как исследование «объектно-реляционных» технологий.

Стоунбрейкер и его аспиранты активно развивали Postgres в течение восьми лет. В течение этого времени Postgres вводили правила, процедуры, расширяемые типы с индексами и объектно-реляционными концепциями. Postgres позже был коммерциализирован, чтобы стать Illustra, который позже был куплен Informix и интегрирован в свой Universal Server. Informix был приобретен IBM в 2001 году на один миллиард долларов.

В 1995 году два кандидата наук. Студенты из лаборатории Стоунбрейкера, Эндрю Ю и Джоли Чэн, заменили язык запросов Postgres POSTQUEL расширенным подмножеством SQL. Они переименовали систему в Postgres95.

В 1996 году Postgres95 отошла от академических кругов и начала новую жизнь в мире с открытым исходным кодом, когда группа преданных разработчиков за пределами Беркли увидела работу этой системы и посвятила себя ее дальнейшему развитию. Содействуя огромному количеству времени, навыкам, труду и технической экспертизе, эта глобальная группа развития радикально трансформировала Postgres. В течение следующих восьми лет они обеспечивали согласованность и единообразие базы кода, создавали подробные регрессионные тесты для обеспечения качества, настраивали списки рассылки для отчетов об ошибках, фиксировали неисчислимые ошибки, добавляли невероятные новые функции и дополняли систему, заполняя различные пробелы таких как документация для разработчиков и пользователей.

Результатом их труда стала новая база данных, получившая репутацию и солидную стабильность. С началом своей новой жизни в мире с открытым исходным кодом, с множеством новых функций и улучшений, система базы данных заняла свое текущее имя: PostgreSQL. («Postgres» по-прежнему используется как старое прозвище начальной системы).

PostgreSQL начинался с версии 6.0, давая кредит на многолетнюю предшествующую разработку. С помощью сотен разработчиков со всего мира система была изменена и улучшена практически во всех областях. В течение следующих четырех лет (версии 6.0 - 7.0) были сделаны значительные улучшения и новые возможности, такие как:

* Мультиверсионный контроль параллелизма. Блокировка на уровне таблиц была заменена сложной системой управления параллельными параллелизмами, которая позволяет читателям продолжать чтение согласованных данных во время активности записи и позволяет делать резервные копии во время работы базы данных в режиме реального времени;
* Важные функции SQL. Было сделано много усовершенствований SQL, включая подзапросы, значения по умолчанию, ограничения, первичные ключи, вторичные ключи, цитируемые идентификаторы, принудительное строковое кодирование, распределение типов ролей, двоичные и шестнадцатеричные целочисленные значения, и другие.
* Улучшены встроенные типы. Были добавлены новые типы, включая широкий диапазон типов дата-время и дополнительных геометрических типов.
* Скорость. Большой прирост производительности размере 20-40 процентов, а время запуска бэкэнда было уменьшено на 80 процентов.

За последующие четыре года (версии с 7.0 по 7.4) создано: журнал опережающей записи (WAL), схемы БД, подготовленные запросы, внешние соединения, сложные запросы, синтаксис join стандарта SQL92, методика хранения сверхбольших атрибутов (TOAST), поддержка IPv6 протокола, стандартную информационную схему SQL, полнотекстовое индексирование, автоматическое вакуумирование, процедурные языки Perl / Python / TCL, улучшенная поддержка SSL, оптимизация SQL запросов, информация о статистике базы данных, добавлена безопасность, функции таблицы и усовершенствования ведения журнала, а также значительные улучшения скорости.

Сегодня пользовательская база PostgreSQL больше, чем когда-либо, и она включена значительную группу крупных корпораций, которые используют ее в сложных условиях. Некоторые из этих компаний, Afilias и Fujitsu, внесли значительный вклад в развитие PostgreSQL. PostgreSQL продолжает улучшаться как в изощренности, так и в производительности более чем когда-либо. Версия 8.0 - долгожданный дебют PostgreSQL на рынке корпоративных баз данных, который включает такие функции, как табличные пространства, хранимые процедуры Java, восстановление времени и вложенные транзакции (точки сохранения). С ним появилась долгожданная функция - портирования Windows.

Многие организации, государственные учреждения и компании используют PostgreSQL.

Многие продолжают говорить о PostgreSQL как о "Postgres" (часто полностью заглавными буквами) потому что так привыкли или потому что так легче произносится. Это название широко используется в качестве сокращения или варианта полного названия PostgreSQL.

При разработке Postgres95 акцент ставился на обнаружение и понимание существующих проблем в коде сервера. В PostgreSQL акцент сместился на расширение возможностей и совместимости при продолжении работы во всех других областях.

Благодаря своему открытому коду, в России появилась своя система на базе PostgreSQL с именем PostgreSQL Pro. Postgres Pro - Российская СУБД, разработанная компанией Postgres Professional на основе свободно-распространяемой СУБД PostgreSQL. Postgres Pro входит в реестр российского ПО. [12]

4.2 PostgreSQL

PostgreSQL - это мощная система объектно-реляционных баз данных с открытым исходным кодом. Она имеет более чем 20-летнюю активную разработку и проверенную архитектуру, которая заслужила высокую репутацию за надежность, целостность данных и правильность. PostgreSQL работает на всех основных операционных системах, включая Linux, UNIX (AIX, BSD, HP-UX, SGI IRIX, macOS, Solaris, Tru64) и Windows, а также полностью совместим с ACID, имеет полную поддержку внешних ключей, соединений, просмотров, триггеров и хранимых процедур (на нескольких языках). Она включает большинство типов данных SQL: типы данных 2008 года, включая INTEGER, NUMERIC, BOOLEAN, CHAR, VARCHAR, DATE, INTERVAL и TIMESTAMP. А также поддерживает хранение двоичных больших объектов, включая изображения, звуки или видео. Имеет встроенные программные интерфейсы для C / C ++, Java, .Net, Perl, Python, Ruby, Tcl, ODBC, среди прочих, и исключительную документацию.

Сильными сторонами PostgreSQL считаются:

* высокопроизводительные и надёжные механизмы транзакций и репликации;
* расширяемая система встроенных языков программирования: в стандартной поставке поддерживаются PL/pgSQL, PL/Perl, PL/Python и PL/Tcl;
* дополнительно можно использовать PL/Java, PL/PHP, PL/Py, PL/R, PL/Ruby, PL/Scheme, PL/sh и PL/V8, а также имеется поддержка загрузки C-совместимых модулей;
* наследование;
* легкая расширяемость.

PostgreSQL создана на основе некоммерческой СУБД Postgres, разработанной как open-source проект в Калифорнийском университете в Беркли. К разработке Postgres, начавшейся в 1986 году, имел непосредственное отношение Майкл Стоунбрейкер, руководитель более раннего проекта Ingres, на тот момент уже приобретённого компанией Computer Associates. Название расшифровывалось как «Post Ingres», и при создании Postgres были применены многие уже ранее сделанные наработки.

PostgreSQL поддерживает большой набор встроенных типов данных:

* численные типы
* целые
* с фиксированной точкой
* с плавающей точкой
* денежный тип (отличается специальным форматом вывода, а в остальном аналогичен числам с фиксированной точкой с двумя знаками после запятой);
* символьные типы произвольной длины;
* двоичные типы (включая blob);
* типы «дата/время» (полностью поддерживающие различные форматы, точность, форматы вывода, включая последние изменения в часовых поясах);
* булев тип;
* перечисление;
* геометрические примитивы;
* сетевые типы;
* ip и ipv6-адреса;
* cidr-формат;
* mac-адрес;
* uuid-идентификатор;
* xml-данные;
* массивы;
* json;
* идентификаторы объектов бд;
* псевдотипы.

Более того, пользователь может самостоятельно создавать новые требуемые ему типы и программировать для них механизмы индексирования с помощью GiST.

PostgreSQL может быть расширен пользователем для собственных нужд практически в любом аспекте. Есть возможность добавлять собственные:

* преобразования типов;
* типы данных;
* домены (пользовательские типы с изначально наложенными ограничениями);
* функции (включая агрегатные);
* индексы;
* операторы (включая переопределение уже существующих);
* процедурные языки.

4.3 Сравнение PostgreSQL c другими СУБД

Для проведения сравнения приведены такие системы управления базами данных которые основаны на открытом исходном коде:

1. SQLite;
2. MySQL;
3. PostgreSQL.

1. SQLite представляет собой библиотеку, скомпонованную вместе с использующим ее приложением. Как и все встраиваемые базы данных, SQLite предоставляет внушительный набор инструментов для работы с самыми разными наборами данных при уменьшенных накладных расходах сравнительно с базами данных, построенных по клиент-серверной модели.

Благодаря тому, что вся база данных хранится в единственном файле, SQLite представляет собой очень быструю и эффективную СУРБД, которая используется как во встраиваемых системах, так и на выделенных машинах.

*Преимущества SQLite.*

* Вся база состоит из единственного файла данных, что обеспечивает высочайший уровень переносимости;
* Несмотря на свою облегченную структуру, в основе SQLite лежит SQL и его стандарты;
* SQLite удобна для разработчиков приложений, поскольку состоит всего из одного файла, а ее библиотека написана на Си и обладает множеством привязок к другим языкам программирования.

*Недостатки SQLite.*

* Отсутствие разграничения прав доступа
* Невозможность использования в крупных проектах
* Невозможность повышения эффективности

*В каких случаях следует использовать SQLite.*

* Для встроенных приложений
* Для прямого доступа к диску
* В целях тестирования

*В каких случаях не использовать SQLite.*

* При проектировании многопользовательских приложений
* При записи больших объемов данных

2. MySQL – самая популярная база данных для серверов с высокой нагрузкой. Это многофункциональный программный продукт с открытым кодом, взаимодействующий с огромным числом сайтов и веб-приложений.

Кроме того, MySQL характеризуется сравнительно низким порогом вхождения.

Несмотря на неполную имплементацию стандартов SQL, MySQL может похвастаться очень высоким функционалом.

MySQL, в отличие от SQLite, реализована согласно спецификациям клиент-серверной архитектуры, поэтому приложения обращаются к базе данных посредством службы MySQL.

*Преимущества MySQL.*

* Легкость установки и наличие визуальных средств управления;
* Поддержка, прямо или косвенно, обширного SQL-функционала;
* Наличие встроенных средств безопасности;
* Масштабируемость и работа с большими объемами данных;
* Игнорирование определенных стандартов увеличивает скорость работы и производительность MySQL.

*Недостатки MySQL.*

* MySQL далеко не универсальна и обладает определенными ограничениями функционала, которые «нравятся» далеко не всем приложениям;
* MySQL не настолько надежна, как хотелось бы, по сравнению с другими СУРБД;
* Несмотря на то, что технически MySQL представляет собой программное обеспечение с открытым кодом, процесс ее разработки фактически прекращен.

*В каких случаях использовать MySQL.*

* В случае необходимости совершения большого количества распределенных транзакций;
* Реализации требований высокой надежности;
* Повышенных требований к гибкости и универсальности для работы с сайтами и веб-приложениями;
* В случае соответствия требованиям индивидуального проекта.

*В каких случаях не использовать MySQL.*

* Если существует необходимость точного соответствия стандартам SQL;
* При необходимости многопоточности и параллелизма;
* При повышенных требованиях к функционалу – например, необходимости наличия полноценного текстового поиска.

3. PostgreSQL – это наиболее продвинутая объектно-реляционная база данных. PostgreSQL ставит перед собой задачу полного соответствия как стандартам SQL, так и международным стандартам ANSI/ISO.

Отличием PostgreSQL от других систем управления базами данных является интеграция объектно-ориентированной и/или реляционной моделей.

Благодаря своей внутренней структуре, PostgreSQL обладает высоким уровнем параллелизма и впечатляющей производительностью.

Несмотря на то, что PostgreSQL не настолько популярна, как MySQL, существует множество приложений сторонних производителей, значительно облегчающих работу с этой чрезвычайно мощной СУБД.

*Преимущества PostgreSQL.*

* PostgreSQL – это ПО с открытым исходным кодом, отвечающее всем стандартам СУРБД;
* Обширное сообщество пользователей;
* Наличие большого числа приложений сторонних производителей для настройки и управления;
* Расширение функционала за счет создания хранимых процедур;
* PostgreSQL – не реляционная, а объектно-реляционная база данных, поддерживающая все свойства ООП.

*Недостатки PostgreSQL.*

* Неудовлетворительная производительность на простых операциях чтения;
* PostgreSQL не пользуется особой популярностью, несмотря на развитое сообщество;
* Мало хостеров, предоставляющих поддержку PostgreSQL.

*Когда следует использовать PostgreSQL.*

* В случае, когда приоритетом является надежность хранения данных и их целостность;
* В случае необходимости выполнения большого количества пользовательских процедур;
* При запланированной миграции на платные СУРБД (например, Oracle);
* При наличии сложных структур данных.

*Когда не нужно использовать PostgreSQL.*

* Скорость чтения не относится к сильным сторонам PostgreSQL. Если данное условие существует, то PostgreSQL – не самый лучший выбор;
* Настройка PostgreSQL – утомительное занятие. Если многие из ее возможностей вам не нужны, то лучше подыскать другую СУРБД;
* В PostgreSQL существуют определенные сложности с репликацией, поэтому, если вы к ним не готовы, лучше использовать MySQL.

Исходя из данного сравнения для реализации системы человеко-машинного взаимодействия в виде бота для мессенджера telegram и делая упор на сильные стороны СУБД, PostgreSQL является отличным решением в данной работе. Так как необходима повышенная целость данных и отличная интеграция в среде Python. Не маловажно использование данной БД для подключения большого числа пользователей.

5 Среда разработки.

В связи с наблюдаемым в настоящее время стремительным развитием персональной вычислительной техники, происходит постепенное изменение требований, предъявляемых к языкам программирования. Все большую роль начинают играть интерпретируемые языки, поскольку возрастающая мощь персональных компьютеров начинает обеспечивать достаточную скорость выполнения интерпретируемых программ. А единственным существенным преимуществом компилируемых языков программирования является создаваемый ими высокоскоростной код. Когда скорость выполнения программы не является критичной величиной, наиболее правильным выбором будет интерпретируемый язык, как более простой и гибкий инструмент программирования.

В связи с этим, определенный интерес представляет рассмотрение сравнительно нового языка программирования Python, который был создан его автором Гвидо ван Россумом в начале 90-х годов.

Отличительные характеристики языка:

− очень низкий порог вхождения, уже после одного дня изучения можно начать писать простые программы;

− минималистичный язык, с небольшим количеством конструкций;

− краткий код;

− прекрасно подходит для создания программ-обёрток, поддерживается импорт Си-библиотек;

− существует большое количество реализаций: CPython (основная реализация); Jython (реализация для JVM); IronPython (CLR); PyPy;

− очень хорошая поддержка математических вычислений (библиотеки NumPy, SciPy);

− используется для обработки естественных языков (NLTK);

− большое количество развитых web-фреймворков (Django, TurboGear, CherryPy, Flask).

Язык можно рекомендовать всем, кто только начинает программировать, как первый язык программирования в жизни.

5.1 Причины использования Python

Python - интерпретируемый язык программирования. С одной стороны, это позволяет значительно упростить отладку программ, с другой - обуславливает сравнительно низкую скорость выполнения.

Динамическая типизация. В python не надо заранее объявлять тип переменной, что очень удобно при разработке.

Хорошая поддержка модульности. Легко написать свой модуль и использовать его в других программах.

Встроенная поддержка Unicode в строках. В Python необязательно писать всё на английском языке, в программах вполне может использоваться родной язык.

Поддержка объектно-ориентированного программирования. При этом его реализация в python является одной из самых понятных.

Автоматическая сборка мусора, отсутствие утечек памяти.

Интеграция с C/C++, если возможностей python недостаточно.

Понятный и лаконичный синтаксис, способствующий ясному отображению кода. Удобная система функций позволяет при грамотном подходе создавать код, в котором будет легко разобраться другому человеку в случае необходимости. Также вы сможете научиться читать программы и модули, написанные другими людьми.

Огромное количество модулей, как входящих в стандартную поставку Python 3, так и сторонних. В некоторых случаях для написания программы достаточно лишь найти подходящие модули и правильно их скомбинировать. Таким образом, вы можете думать о составлении программы на более высоком уровне, работая с уже готовыми элементами, выполняющими различные действия.

Кроссплатформенность. Программа, написанная на Python, будет функционировать совершенно одинаково вне зависимости от того, в какой операционной системе она запущена. Отличия возникают лишь в редких случаях, и их легко заранее предусмотреть благодаря наличию подробной документации.

Качество программного обеспечения.

Для многих основное преимущество языка Python заключается в удобочитаемости, ясности и более высоком качестве, отличающими его от других инструментов в мире языков программирования. Программный код на языке Python читается легче, а значит, многократное его использование и обслуживание выполняется гораздо проще, чем использование программного кода на других языках сценариев. Единообразие оформления программного кода на языке Python облегчает его понимание даже для тех, кто не участвовал в его создании. Кроме того, Python поддерживает самые современные механизмы многократного использования программного кода, каким является объектно-ориентированное программирование (ООП).

Высокая скорость разработки.

По сравнению с компилирующими или строго типизированными языками, такими как C, C++ и Java, Python во много раз повышает производительность труда разработчика. Объем программного кода на языке Python обычно составляет треть или даже пятую часть эквивалентного программного кода на языке C++ или Java. Это означает меньший объем ввода с клавиатуры, меньшее количество времени на отладку и меньший объем трудозатрат на сопровождение. Кроме того, программы на языке Python запускаются сразу же, минуя длительные этапы компиляции и связывания, необходимые в некоторых других языках программирования, что еще больше увеличивает производительность труда программиста.

Переносимость программ.

Большая часть программ на языке Python выполняется без изменений на всех основных платформах. Перенос программного кода из операционной системы Linux в Windows обычно заключается в простом копировании файлов программ с одной машины на другую. Более того, Python предоставляет массу возможностей по созданию переносимых графических интерфейсов, программ доступа к базам данных, веб-приложений и многих других типов программ. Даже интерфейсы операционных систем, включая способ запуска программ и обработку каталогов, в языке Python реализованы переносимым способом.

Библиотеки поддержки.

В составе Python поставляется большое число собранных и переносимых функциональных возможностей, известных как стандартная библиотека. Эта библиотека предоставляет массу возможностей, востребованных в прикладных программах, начиная от поиска текста по шаблону и заканчивая сетевыми функциями. Кроме того, Python допускает расширение как за счёт ваших собственных библиотек, так и за счет библиотек, созданных сторонними разработчиками. Из числа сторонних разработок можно назвать инструменты создания веб-сайтов, программирование математических вычислений, доступ к последовательному порту, разработку игровых программ и многое другое. Например, расширение NumPy позиционируется как свободный и более мощный эквивалент системы программирования математических вычислений Mathlab.

Интеграция компонентов.

Сценарии Python легко могут взаимодействовать с другими частями приложения благодаря различным механизмам интеграции. Эта интеграция позволяет использовать Python для настройки и расширения функциональных возможностей программных продуктов. На сегодняшний день программный код на языке Python имеет возможность вызывать функции из библиотек на языке C/C++, сам вызываться из программ, написанных на языке C/C++, интегрироваться с программными компонентами на языке Java, взаимодействовать с такими платформами, как COM и .NET, и производить обмен данными через последовательный порт или по сети с помощью таких протоколов, как SOAP, XML-RPC и CORBA.

5.2 История создания

Разработка языка Python была начата в конце 1980-х годов сотрудником голландского института CWI Гвидо ван Россумом. Для распределённой ОС Amoeba требовался расширяемый скриптовый язык, и Гвидо начал писать Python на досуге, позаимствовав некоторые наработки для языка ABC (Гвидо участвовал в разработке этого языка, ориентированного на обучение программированию). В феврале 1991 года Гвидо опубликовал исходный текст в ньюсгруппе alt.sources. С самого начала Python проектировался как объектно-ориентированный язык.

Название языка произошло вовсе не от вида пресмыкающихся. Автор назвал язык в честь популярного британского комедийного телешоу 1970-х "Летающий цирк Монти Пайтона". Впрочем, всё равно название языка чаще ассоциируют именно со змеёй, нежели с передачей — пиктограммы файлов в KDE или в Microsoft Windows и даже эмблема на сайте python.org (до выхода версии 2.5) изображают змеиные головы.

Наличие дружелюбного, отзывчивого сообщества пользователей считается наряду с дизайнерской интуицией Гвидо одним из факторов успеха Python. Развитие языка происходит согласно чётко регламентированному процессу создания, обсуждения, отбора и реализации документов PEP (англ. Python Enhancement Proposal) — предложений по развитию Python.

3 декабря 2008 года, после длительного тестирования, вышла первая версия Python 3000 (или Python 3.0, также используется сокращение Py3k). В Python 3000 устранены многие недостатки архитектуры с максимально возможным (но не полным) сохранением совместимости со старыми версиями Python. На сегодня поддерживаются обе ветви развития (Python 3.x и 2.x)

5.3 Влияние других языков

Появившись сравнительно поздно, Python создавался под влиянием множества языков программирования:

− ABC — отступы для группировки операторов, высокоуровневые структуры данных (map) (Python фактически создавался как попытка исправить ошибки, допущенные при проектировании ABC);

− Modula-3 — пакеты, модули, использование else совместно с try и except, именованные аргументы функций (на это также повлиял Common Lisp);

− С, C++ — некоторые синтаксические конструкции (как пишет сам Гвидо ван Россум — он использовал наиболее непротиворечивые конструкции из С, чтобы не вызвать неприязнь у С-программистов к Python);

− Smalltalk — объектно-ориентированное программирование;

− Lisp — отдельные черты функционального программирования (lambda, map, reduce, filter и другие);

− Fortran — срезы массивов, комплексная арифметика;

− Miranda — списочные выражения;

− Java — модули logging, unittest, threading (часть возможностей оригинального модуля не реализована), xml.sax стандартной библиотеки, совместное использование finally и except при обработке исключений, использование @ для декораторов;

− Icon — генераторы.

Большая часть других возможностей Python (например, байт-компиляция исходного кода) также была реализована ранее в других языках

5.4 Преимущество Python перед другими языками высокого уровня

− Имеет более широкие возможности, чем Tcl. Язык Python поддерживает "программирование в целом", что делает его применимым для разработки крупных систем.

− Имеет более четкий синтаксис и более простую архитектуру, чем Perl, что делает программный код более удобочитаемым, простым в сопровождении и снижает вероятность появления ошибок.

− Проще и удобнее, чем Java. Python – это язык сценариев, а Java унаследовала сложный синтаксис от таких языков программирования, как C++.

− Проще и удобнее, чем C++, но нередко он не может конкурировать с C++, поскольку, будучи языком сценариев, Python предназначен для решения другого круга задач.

− Более мощный и более переносимый, чем Visual Basic. Открытая природа Python также означает, что нет какой-то отдельной компании, которая его контролирует.

− Более удобочитаемый и более универсальный, чем PHP. Иногда Python используется для создания веб-сайтов, но он способен решать гораздо более широкий круг задач, от управления роботами до создания анимационных фильмов.

− Более зрелый и имеет более ясный синтаксис, чем Ruby. В отличие от Ruby и Java, объектно-ориентированный стиль программирования является необязательным в Python – он не вынуждает использовать ООП в проектах, где этот стиль неприменим.

Обладает динамическими особенностями таких языков, как SmallTalk и Lisp, но имеет более простой и традиционный синтаксис, доступный как для разработчиков, так и для конечных пользователей настраиваемых систем.

5.5 Реализации Python

CPython является основной, но не единственной реализацией языка программирования Python. Существуют также следующие реализации:

* Jython — реализация Python, использующая JVM в качестве среды исполнения. Позволяет прозрачно использовать Java-библиотеки;
* PyS60 — реализация языка для смартфонов фирмы Nokia на платформе Series 60;
* IronPython — Python для .NET Framework и Mono. Компилирует Python программы в MSIL, таким образом предоставляя полную интеграцию с .NET-системой;
* Stackless — также написанная на Си реализация Python. Это не полноценная реализация, а патчи к CPython. Предоставляет расширенные возможности многопоточного программирования и значительно большую глубину рекурсии;
* Python for .NET — ещё одна реализация Python для .NET. В отличие от IronPython эта реализация не компилирует Python код в MSIL, а только предоставляет интерпретатор, написанный на C#. Позволяет использовать .NET-сборки из Python кода;
* PyPy — реализация Python, написанная на Python. Позволяет легко проверять новые возможности. В PyPy кроме стандартного CPython включены возможности Stackless, Psyco. В проект интегрированы возможности анализа Python кода и трансляция в другие языки и байткоды виртуальных машин;
* PyCharm – это самая интеллектуальная Python IDE с полным набором средств для эффективной разработки на языке Python. Выпускается в двух вариантах – бесплатная версия PyCharm Community Edition и поддерживающая больший набор возможностей PyCharm Professional Edition. PyCharm выполняет инспекцию кода на лету, автодополнение, в том числе основываясь на информации, полученной во время исполнения кода, навигацию по коду, обеспечивает множество рефакторингов. В данной работ используется именно эта реализация для программирования на Python. Так в ней очень удобно инспектировать и править код программы и остальные плюсы, описанные выше.

5.6 Недостатки

Низкое быстродействие. Классический Python, как и многие другие интерпретируемые языки, не применяющие, например, JIT-компиляторы, имеют общий недостаток — сравнительно невысокую скорость выполнения программ. Сохранение байт-кода позволяет интерпретатору не тратить лишнее время на перекомпиляцию кода модулей при каждом запуске, в отличие, например, от языка Perl. Кроме того, существует специальная JIT-библиотека psyco, позволяющая ускорить в выполнение программ (однако приводящая к увеличению потребления оперативной памяти). Эффективность psyco сильно зависит от архитектуры программы.

Существуют реализации языка Python, вводящие высокопроизводительные виртуальные машины (ВМ) в качестве бэк-энда компилятора. Примерами таких реализаций может служить PyPy, базирующийся на LLVM; более ранней инициативой является проект Parrot. Ожидается, что использование ВМ типа LLVM приведёт к тем же результатам, что и использование аналогичных подходов для реализаций языка Java, где низкая вычислительная производительность в основном преодолена.

Множество программ/библиотек для интеграции с другими языками программирования предоставляют возможность использовать другой язык для написания критических участков.

Невозможность модификации встроенных классов. По сравнению с Ruby и некоторыми другими языками, в Python отсутствует возможность модифицировать встроенные классы, такие, как int, str, float, list и другие, что, однако, позволяет Python потреблять меньше оперативной памяти и быстрее работать. Ещё одной причиной введения такого ограничения является необходимость согласования с модулями расширения. Многие модули (в целях оптимизации быстродействия) преобразуют Python-объекты элементарных типов к соответствующим Си-типам вместо манипуляций с ними посредством Си-API. Также это избавляет от многих потенциальных ошибок при неконтролируемом динамическом переопределении встроенных типов.

Для реализации человеко-машинного взаимодействия, данная особенность низкого быстродействия не сильно скажется на работе бота информера, так как на данном этапе разработки, система не подразумевает большого количества пользователей, что значит бот сможет давать ответ пользователю за коротки промежуток времени. Тем не менее такой недостаток не сильно скажется на пользователе, чем на разработчике.

6 Мессенджер Telegram

6.1 Основные причины выбора данного мессенджера

Для создания канала человеко-машинного взаимодействия необходимо использовать в конкретно данной области подходящий мессенджер, в котором есть необходимые качества для реализации работы. Просмотр всех существующих мессенджеров таких как WhatsApp, Viber и др. заставил остановиться именно на Telegram. Описание полученных качеств приведены ниже.

1. Скорость работы. В telegram скорость отправки сообщений моментальная, даже при низкой скорости интернета. Все передается за секунды, да и сервера падают крайне редко. Это очень большой плюс для данного мессенджера.
2. Дизайн. Он очень эстетичный, все подогнано под один стиль, ни что не выделяется. По канонам Material Design, быстро и в минимальное количество кликов. Так же плюсом является размер приложения, он меньше чем у Whatsapp, Viber and etc.
3. Всеядность. Абсолютно везде применим данный мессенджер – Android, iOS, Windows, Mac OS и даже Web-клиент. Причем работает одинаково хорошо во всех платформах.
4. Облачность. Все переписки, фото, видео, хранятся в облаке. Все данные сжаты(урезаны), но они не потеряются. Это конечно не ново, но реализовано лучше, чем у WhatsApp, где вас спрашивают сохранять или нет. В данном случае, здесь всё реализовано по стандарту и ничего не пропадёт.
5. Безопасность. Секретные чаты с авто удалением, пароли на вход и двухэтапная авторизация. То есть, он уже более защищен чем остальные, и это может стать решающим фактором для многих, но работает все как часы, пароль никак не обойти, двухэтапная авторизация при попытке взлома.
6. Боты, каналы и стикеры. Возможность создания бота с помощью клиента мессенджера, которого можно использовать по своему усмотрению. Создание большого количества каналов для общения и делиться информацией в виде массовой рассылки. Ещё огромное количество разнообразных стикеров, которые доступны всем бесплатно, а также можно добавлять свои стикеры в коллекцию.

Для реализации цели данной магистерской работы, является возможность создания бота в мессенджере telegram.

Telegram — бесплатный кроссплатформенный мессенджер для смартфонов и других устройств, позволяющий обмениваться текстовыми сообщениями и медиафайлами различных форматов. Используются проприетарная серверная часть c закрытым кодом, работающая на мощностях нескольких компаний США и Германии, финансируемых Павлом Дуровым в объёме порядка 12 млн долларов США ежегодно, и несколько клиентов с открытым исходным кодом, в том числе под GNU GPL.

Количество активных пользователей сервиса на февраль 2016 года составляло более 100 млн человек, а количество ежедневно пересылаемых сообщений достигло 10 миллиардов на август 2015.

6.2 История

Проект создан Павлом Дуровым, основателем социальной сети «ВКонтакте». В интервью The New York Times Павел рассказал, что первоначальная идея приложения пришла ему ещё в 2011-м, когда к его двери приходили спецназовцы. Когда последние всё-таки ушли, Дуров сразу же написал своему брату Николаю. Тогда же он и осознал, что у него нет безопасного способа коммуникации с братом. Сервис построен на технологии шифрования переписки MTProto, разработанной братом Павла — Николаем. Сам «Телеграм» изначально был экспериментом принадлежащей Павлу компании Digital Fortress с целью протестировать MTProto на больших нагрузках.

* 14 августа 2013 года представлен первый клиент Telegram для устройств на платформе iOS.
* 22 августа 2013 года один из участников конкурса Durov’s Android Challenge написал и выложил в открытый доступ первое приложение для операционной системы Android, совместимое с Telegram, (использует тот же протокол MTProto).
* В октябре у проекта открылся веб-сайт и была представлена официальная версия Telegram под Android с открытым исходным кодом (GPL2). Предыдущая версия программы доступна под названием «Unofficial Telegram S».
* 7 ноября 2013 года появились сторонние клиенты сервиса для Windows и OS X с ограниченным функционалом. Также был разработан концепт веб-версии клиента.
* В ноябре у программы насчитывалось, по данным «TJournal», около 1 миллиона установок.
* В январе 2014 года вышла неофициальная веб-версия Webogram от бывшего разработчика ВКонтакте Игоря Жукова.
* 21 июля 2014 года в App Store появилось приложение Telegram HD для iPhone и iPad, которое загрузила компания Telegram Messenger LLP.
* Новое приложение получило специальную версию для Apple iPad, улучшило поддержку видео и фотографий высокого разрешения, добавило возможность пересылки анимированных изображений в формате gif. На официальном сайте мессенджера в качестве клиента для iOS указано именно это приложение.
* 15 октября 2014 года в Telegram была добавлена поддержка псевдонимов, по которым возможно связываться с пользователями, даже не зная их телефонного номера, а также запущен веб-клиент.
* 2 января 2015 года в Telegram была добавлена поддержка стикеров. Изначально в приложении 14 стикеров, но любой пользователь может модифицировать их или добавить свои собственные. В отличие от многих приложений, в Telegram стикеры полностью бесплатные.
* В феврале 2016 года один из создателей Telegram Павел Дуров заявил, что мессенджером пользуются уже более 100 миллионов человек, при этом сервис доставляет около 15 миллиардов сообщений ежедневно. Еще в сентябре 2015 года Telegram передавал 12 миллиардов посланий в день.
* В апреле 2016 года стало известно, что в мае 2015 года корпорация Google рассматривала возможность покупки мессенджера за более чем 1 млрд долларов США.
* В мае 2016 года появилась возможность редактирования отправленных сообщений. Внести изменения возможно в течение двух суток с момента отправки. В этом случае в сообщении появится специальная метка.
* 22 ноября 2016 года разработчиками был запущен проект Telegraph — блог-платформа, бесплатный издательский инструмент, который позволяет создавать публикации, обзоры, вставлять фотографии и все виды embedded-кода. Telegraph — гибрид блог-платформы, мессенджера и платишера (подобного Medium), с концепцией анонимных имиджборд. Проект Павла Дурова, основателя социальной сети «ВКонтакте»
* 3 января 2017 года один из разработчиков добавил возможность удаления своих отправленных сообщений. После удаления отправителем сообщения, собеседник не сможет увидеть удаленное сообщение.
* В марте 2017 года В. Д. Соловей со ссылкой на анонимный источник сообщил, что российские спецслужбы получили доступ к сообщениям пользователей и их архиву за три года. Павел Дуров назвал это заявление уткой.
* 15 мая 2017 года стало известно о том, что декстопная версия Telegram получила возможность совершать звонки.
* 16 мая 2017 года администрация Telegram заявила, что не будет предоставлять информацию российским госорганам.

6.3 Технология

Для мессенджера был создан протокол MTProto, предполагающий использование нескольких протоколов шифрования. При авторизации и аутентификации используются алгоритмы RSA-2048, DH-2048 для шифрования, при передаче сообщений протокола в сеть они шифруются AES с ключом, известным клиенту и серверу. Также применяются криптографические хеш-алгоритмы SHA-1 и MD5.

Безопасность от перехвата пересылаемых сообщений со стороны сервера Telegram обеспечивается лишь в режиме «секретных» чатов (Secret Chats), доступном с 8 октября 2013 года. Этот режим реализует шифрование, при котором лишь отправитель и получатель обладают общим ключом (end-to-end шифрование), с применением алгоритма AES-256 в режиме IGE (англ. Infinite Garble Extension) для пересылаемых сообщений. В отличие от обычного режима, сообщения в секретных чатах не расшифровываются сервером, история переписки сохраняется лишь на тех двух устройствах, на которых был создан чат.

Имеется возможность организовывать мультичаты до 200 участников, начиная с ноября 2015 года, супергруппы до 1000 участников, с 14 марта 2016 — супергруппы до 5000 участников.

6.4 Боты

При помощи специального API сторонние разработчики могут создавать «ботов», специальные аккаунты, управляемые программами. Типичные боты отвечают на специальные команды в персональных и групповых чатах, также они могут осуществлять поиск в интернете или выполнять иные задачи, применяются в развлекательных целях или в бизнесе.

В сентябре 2015 года Павел Дуров заявил о скором появлении возможностей монетизации и размещения рекламы в ботах.

6.5 Преимущества и недостатки использования Telegram

В пользу использования этой программы можно отметить следующие характеристики:

* бесплатное пользование приложением на неограниченный срок;
* высококачественное шифрование информации;
* мгновенная доставка пересланных данных;
* отсутствие рекламных блоков и микро-транзакций;
* возможность самостоятельной настройки и регулирования приложения «под себя».

Скромные, но тем не менее, недостатки:

* отсутствие интерфейса с переводом на русский язык;
* исходный код серверной части закрыт для пользователей.

Несмотря на это, Телеграмм остается одним из самых удобных, а главное безопасных способов передачи информации в сети. Ведь разработчики приложения доказали, что обеспечить конфиденциальность в интернете возможно.

6.6 Секретный чат

Мессенджер Telegram обладает некоторыми отличительными положи-тельными чертами, относящимися к безопасности пользования данной программой. Одной из основных черт считается секретный чат в Telegram.

Если подробнее ознакомиться с этой функцией, то можно увидеть ряд особенностей её применения. Переписка не подвергается сохранению на сервере и моментально удаляется сразу после отправления сообщения. Остальные пользователи, естественно, не могут прочитать её. А для повышения мощности защиты реализуется конечное шифрование сообщения. Таким образом, отправленное сообщение остается доступным лишь для его адресата и адресанта. Применяемые системой алгоритмы гарантируют высокий уровень безопасности, а расшифровать чужую личную переписку не могут даже сами разработчики мессенджера.

Для каждого сообщения или отдельной переписки можно подобрать и установить отдельный таймер самоликвидации, который составляет минимум две секунды и максимум семь дней. Как только срок завершается, сообщение самостоятельно удаляется.

Получатель читает новое сообщение и через несколько секунд оно попро-сту исчезает. При проявлении смекалки, конечно, можно успеть сделать снимок экрана или скриншот. Но в таком случае программа зафиксирует скриншот и перешлет отправителю уведомление об этой операции.

6.7 Мессенджеры в России

Согласно аналитическому источнику Brand Analytics по данным мобильных операторов в России мессенджер номер один – WhatsApp, им пользуются 68,7 процентов клиентов Билайна и 47,6 процентов среди абонентов Мегафон. На втором месте Viber – 45,7 процентов и 39,7 процентов соответственно. Доля пользователей Телеграма пока не так существенна – этим мессенджером пользуются всего 7,5 процентов абонентов Билайна.

Анализ обсуждений каналов коммуникаций позволяет выявить какие из них наиболее востребованы, а какие – теряют свои позиции. Наибольшая доля сообщений в феврале 2017 года содержит упоминания WhatsApp – 30 процентов и Viber – 27 процентов. SMS занимают третью строчку рейтинга с долей 16 процентов, у Skype – 14 процентов. Доля упоминаний Телеграма составила 11 процентов от общего объема. В категорию «Другие» попали такие мессенджеры как Snapchat, ICQ, iMessage и Facebook Messenger, доля каждого из которых в сообщениях пользователей социальных медиа составила менее 1 процента от общего объема упоминаний. В соответствии с рисунком 11 представлен рейтинг упоминай мессенджеров за февраль 2017 года.



Рисунок 11 – Рейтинг упоминаний мессенджеров в России за февраль 2017 года

Для сравнения взять данные за март 2016 года по рейтингу мессенджеров и выявить какие теряют свою силу, а какие набирают популярность среди пользователей. В соответствии с рисунком 12 представлен рейтинг упоминай мессенджеров за март 2016 года.



Рисунок 12 – Рейтинг упоминаний мессенджеров в России за март 2016 года

Из данного рейтинга можно отметить, что Skype опустился на четвертое место, WhatsApp занимает первую позицию, Viber вторую, Telegram остался на том же месте, но процент упоминаний возрос до 11.

Абсолютным лидером роста по числу упоминаний по сравнению с данными марта 2016 года стал Телеграм. Этот мессенджер показал четырехкратный рост – со 186 000 до 836 000 упоминаний в месяц.

Наиболее резкое падение объема упоминаний – у Skype – в 1,6 раз по сравнению с прошлым годом и у ICQ – в 2 раза. Последний включен в «другие» данных за февраль 2017 года. В соответствии с рисунком 13, представлен рост месячного объёма упоминаний мессенджеров в России

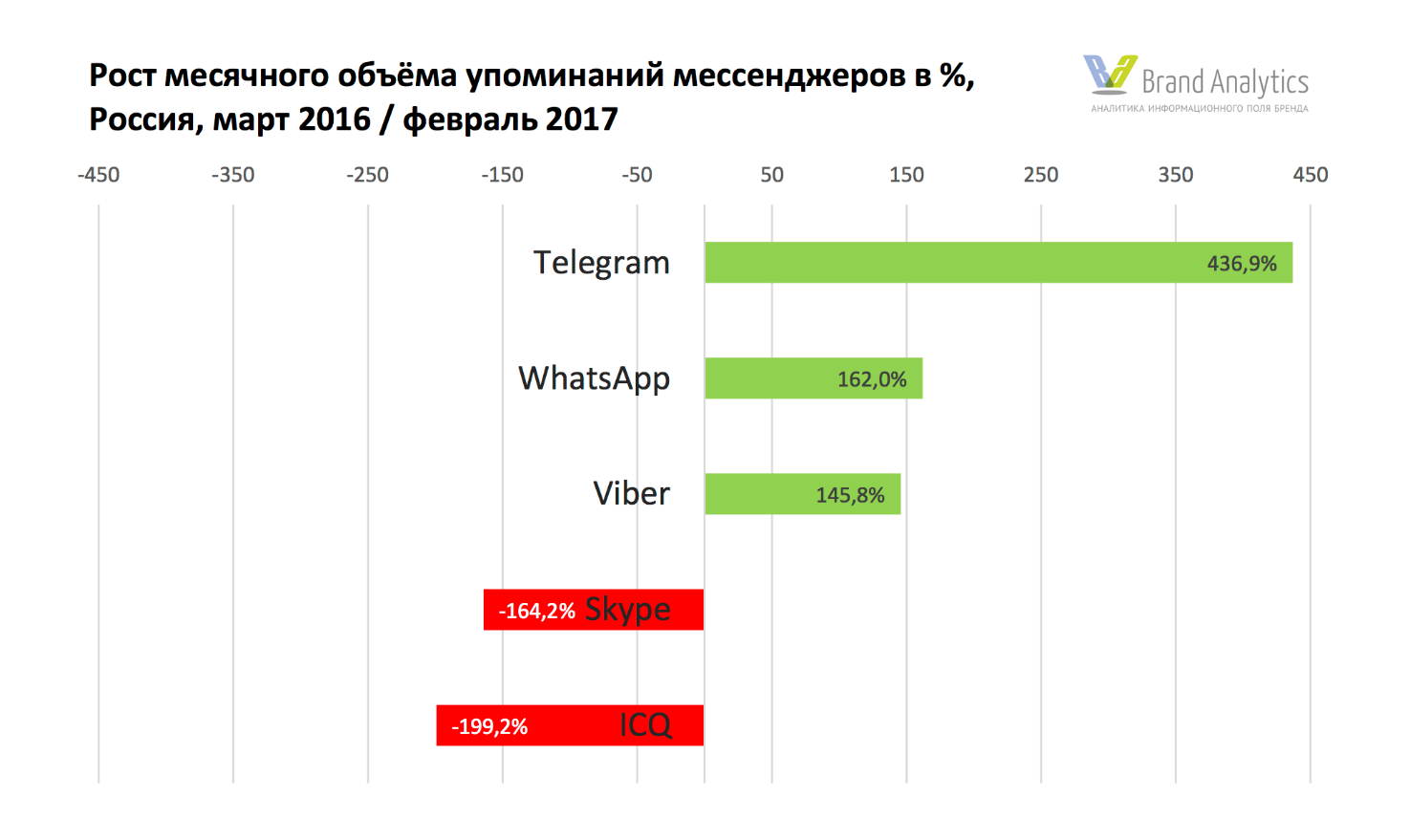


Рисунок 13 – Рост месячного объёма упоминаний мессенджеров в России за март 2016, февраль 2017 гг. в процентах

После представления и анализа полученных данных можно сказать следующее, что использование мессенджеров не только для личного общения, но и для общения в публичных чатах и чтения публичных каналов – один из трендов 2017 года. Чаты и каналы превращают мессенджеры в социальные медиа и привлекают новых пользователей. Тренд-сеттером тут безусловно является Телеграм, что, вероятно, и помогает мессенджеру расти столь высокими темпами.

Так же не маловажно отметить, рост численности пользователей мессенджеров превысил количество пользователей социальных сетей и неуклонно растёт.

7 Моделирование системы поддержки принятия решений при обращении к боту мессенджеру

Создание бота для мессенджера телеграмм состоит из:

1. Регистрации бота через главного бот @botfather (одноразовая операция);
2. Передаётся токен бота для проверки на валидность;
3. Начало работы с ботом с обязательным набором команды /start;
4. Ответ бота на команду /start. Показ интерфейса;
5. Взаимодействие пользователя с ботом по принципу, пользователь задаёт вопрос либо вводит команду;
6. Обработка запроса;
7. Вывод информации пользователю по заданному запросу.

Создание базы данных в PostgreSQL.

В pgAdmin происходит создание базы данных, в соответствии с рисунком 14 и называется question\_answer (вопрос - ответ).

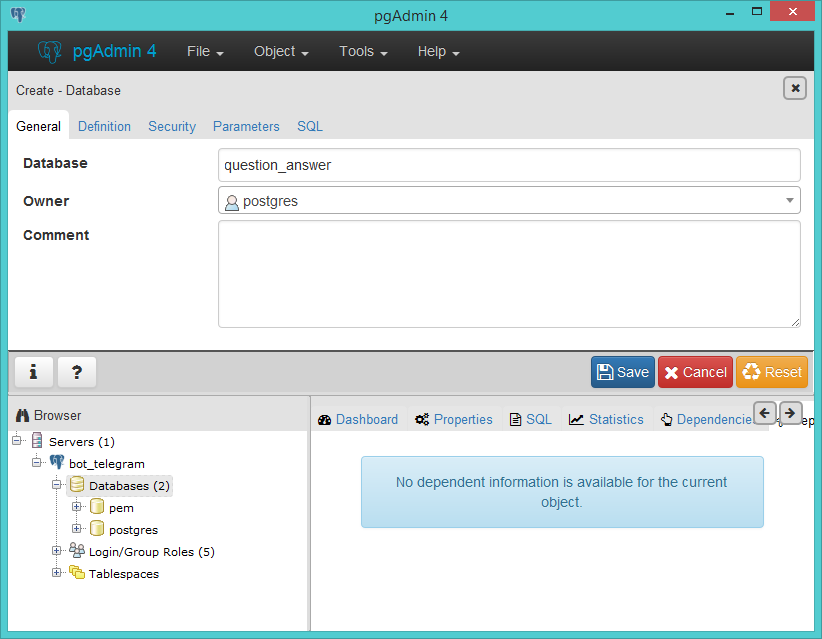


Рисунок 14 – Создание базы данных

Для работы необходимо создать две простые таблицы, одна будет хранить вопросы имя этой таблицы (question), другая будет содержать ответы (answer). С помощью команды *CREATE* создаются эти самые таблицы с указанием типов данных.

Таблица «вопрос».

*CREATE TABLE question (*

*id\_question integer,*

*question text*

*);*

Таблица «ответ».

CREATE TABLE answer (

id\_answer integer,

answer text

);

Добавление данных в таблицу осуществляется с помощью команды *INSERT* или *COPY* (для большого количества записей за раз). Используется команда *INSERT* для добавления записей в таблицу «вопрос».

INSERT INTO question (id\_question, question) VALUES

(1, 'Как дела?'),

(2, 'Какая сейчас погода?'),

(3, 'Кто ты?');

Проверка занесённых данных в таблицу выполняется командой *SELECT \* FROM public.question*, в соответствии с рисунком 15 отображена информация о всех записях в таблице question.

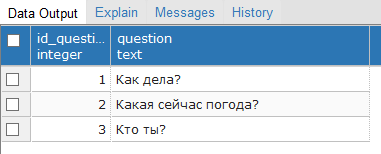


Рисунок 15 – Данные в таблице question

Созданная база данных построена в виде простой схемы. В соответствии с рисунком 16 схема данных представляет собой таблицу с вопросами(question) и таблицу с ответами(answer) имеющую связь один ко многим, то есть у одного вопроса возможно несколько вариантов ответа. Однако стоить отметить, что, если вопрос содержит некую другую специализацию, имеется ввиду, что, если содержится зарезервированное слово, которое активирует подключение к некоторому публичному API, например, «какая сейчас погода?», система определяет слово «погода» и тем самым выводиться информацию по данному запросу, ежели вопрос имеет простую форму вида «как дела?» тогда последует простой ответ «нормально» или какой-либо другой.

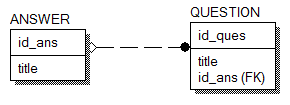


Рисунок 16 – Схема данных

Для увеличения функций бота, создана дополнительная таблица, которая имеет название «JOKE» (анекдот) и в соответствии с рисунком 17 представлена в виде простой таблицы, содержащей идентификатор и поля с данными. Данная таблица хранит анекдоты, которые представлены пользователю по нажатии соответствующей команды и выданы случайно из общего списка.

Для создания новой таблицы используется команда CREATE TABLE JOKE (id\_joke integer, title varchar(255) )

Где id – это номер анекдота с целочисленным типом данных, а JOKE – сама таблица, в которой хранятся сами анекдоты в символьных строках длиной до 255 символов

Для добавления записей в таблицу используется оператор INSERT:

*INSERT INTO JOKE (*id\_joke, title*) VALUES*

*(1, ‘Анекдот 1’),*

*(2, ‘Анекдот 2’);*

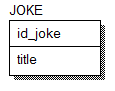


Рисунок 17 – Таблица «Анекдот»

Подключение к среде программирования Python данной базы, происходит с помощью специальной заранее установленной библиотеки *py‑postgresql*.

Первым происходит импорт модуля и делается псевдоним:

*import postgresql.driver as pg\_driver*.

Далее создание подключения к базе:

*db = pg\_driver.connect(user='postgres', password='postgres', host='171.33.252.52', database='question\_answer', port=5432)*.

Подключение к базе завершено, если доступ к базе более не нужен, обязательно использовать параметр закрытия базы *db.close()*.

Все последующие действия по добавлению данных в базу осуществляются через среду программирования Python посредством специальных команд.

Заключение

Системы поддержки принятия решений являются мощными инструментами, объединяющими научные методы поддержки сложных решений с методами, разработанными в области информатики, и становятся все более популярными во многих областях.

Они особенно ценны в ситуациях, когда объем доступной информации является неподъемным для интуиции невостребованного лица, принимающего решения, и в котором важны точность и оптимальность.

В данной работе рассмотрен способ разработки системы поддержки принятия решения в ежедневном быту. Целью работы является создание СППР с интерфейсом в виде бота для мессенджера telegram.

Для достижения цели:

* был проведен анализ СППР, какие изменения коснулись данных систем, применение таких систем, что происходит на рынке и какое дальнейшее развитие;
* описано применение публичные API и так же для чего их использовать в разработке подобной системы в виде бота мессенджера;
* создана и показана БД в СРУБД PostgreSQL
* написан алгоритм работы СППР к каналам человеко-машинного взаимодействия в виде бота для мессенджера telergam;
* создана блок-схема разработанного бота;
* проведена интеграция БД со средой Python;
* написан бот для мессенджера telegram.

Обоснована и доказана актуальность данной работы на основе рынка и развития технологий. Исследование BI-систем наглядно показывает стабильный положительный финансовый прирост рынка в течение 7 лет в среднем на 5-10 процентов, что является хорошей платформой для развития данного направления с целью извлечения максимальной прибыли. При сохранении, либо увеличении данной тенденции можно предположить, что на 2027 год российский рынок BI-систем составит минимально 150 млрд. рублей.

Так же, немало важным является практическая польза для рядовых пользователей различной техники.

Разработанная система человеко-машинного взаимодействия в виде бота для мессенджера telegram значительно облегчает процесс поиска информации, в том числе при содействии рекомендательных сервисов. Все это позволяет охватывать более широкую возрастную аудиторию, а значит и получать еще большее развитие и инвестиции, благодаря простоте и быстроте пользования.

Библиографический список

1. Бабаев, А. А. Системы поддержки принятия решений. [Текст] : учебник и практикум для бакалавриата и магистратуры / А. А. Бабаев, В. Г. Халин, А. В. Юрков; под. ред. В. Г. Халина, Г. В. Черновой. – Москва. : Юрайт, 2015. – 478 с.
2. Ибрар, А. PostgreSQL 9.6 Высокая производительность [Текст] : обучающее руководсто / А. Ибрар, Г. Смит – Издательство ПАКТ: Бирмингем, Великобритания, 2017. – 500 с.
3. Кравченко, Т. Системы поддержки принятия решений [Текст] : учебник и практикум / Т. Кравченко, Д. Исаев – М. : – Юрайт, 2017. – 292 с.
4. Любанович, Б. Простой Python. [Текст] : учебник / Б. Любанович – Издательский дом Питер: Современный стиль программирования, 2016. – 480 с.
5. Bell A. Analytic Hierarchy Process Analyzes Risk of Operating Cross-Country Petroleum Pipelines in India: – Режим доступа: <http://ascelibrary.org/doi/abs/10.1061/%28ASCE%291527-6988%282003%294%3A4%28213%29>. (дата обращения: 26.05.2017). [5]
6. Berrittella M., Certa A., Zito P. An Analytic Hierarchy Process for The Evaluation of Transport Policies to Reduce Climate Change Impacts: – Режим доступа: <http://www.feem.it/userfiles/attach/Publication/NDL2007/NDL2007-012.pdf>. (дата обращения: 20.05.2017). [2]
7. Business Intelligence (рынок России): – Режим доступа: <http://www.tadviser.ru/index.php/Статья:Business_Intelligence_(рынок_России)>. (дата обращения: 04.05.2017). [11]
8. Current weather data: – Режим доступа: <https://openweathermap.org/current>. (дата обращения: 03.06.2017).
9. DSS - система поддержки принятия решений: – Режим доступа: <http://pro-spo.ru/erp/1816-dss>. (дата обращения: 02.03.2017).
10. Duberstein J., Lopes V. The Analytic Hierarchy Process as a Means for Integrated Watershed Management: – Режим доступа: http://www.tucson.ars.ag.gov/icrw/Proceedings/Steiguer.pdf. (дата обращения: 30.05.2017). [6]
11. John R. Improving the Faculty Selection Process in Higher Education: A Case for the Analytic Hierarchy Process: – Режим доступа: <http://files.eric.ed.gov/fulltext/ED504373.pdf>. (дата обращения: 25.05.2017). [4]
12. Locatelli G., Mancini M. A framework for the selection of the right nuclear power plant: – Режим доступа: <http://www.tandfonline.com/doi/abs/10.1080/00207543.2012.657965>. (дата обращения: 23.05.2017). [1]
13. MSDN Magazine Blog: – Режим доступа: https://msdn.microsoft.com/ru-ru/magazine/0117mag. (дата обращения: 19.05.2017). [3]
14. Pavement Management; Monitoring, Evaluation, and Data Storage; and Accelerated Testing 2007: – Режим доступа: <http://pubsindex.trb.org/view/836203>. (дата обращения: 29.05.2017). [7]
15. PostgreSQL: – Режим доступа: <https://ru.wikipedia.org/wiki/PostgreSQL>. (дата обращения: 09.05.2017).
16. Python 3 для начинающих: – Режим доступа: <https://pythonworld.ru>. (дата обращения: 21.05.2017).
17. Python: – Режим доступа: <https://ru.wikipedia.org/wiki/Python>. (дата обращения: 20.05.2017).
18. Telegram (мессенджер): – Режим доступа: <https://ru.wikipedia.org/wiki/Telegram_(мессенджер)>. (дата обращения: 14.05.2017).
19. Telegram Messenger — Новая эра общения: – Режим доступа: <http://telegram-online.ru>. (дата обращения: 11.05.2017).
20. Абраков А. Ш., Сушков Ю. А. Диалоговая система поддержки принятия рациональных решений: – Режим доступа:<http://www.treko.ru/show_article_763>. (дата обращения: 15.04.2017).
21. Анонимный Телеграмм: правда о безопасности виртуального общения: – Режим доступа: <https://telegram-store.com/blog/anonymous-telegram/>. (дата обращения: 10.05.2017).
22. Кадиров Р.М., Рабаданова Р.М., Аналитические информационные системы для поддержки принятия решений: – Режим доступа: <http://www.scienceforum.ru/2017/2464/34008>. (дата обращения: 22.05.2017). [10]
23. Как искать каналы в Телеграмм приложении): – Режим доступа: <http://telegram-free.ru/kak-iskat-kanaly-v-telegram.html>. (дата обращения: 03.05.2017).
24. Краткая история PostgreSQL: – Режим доступа: <http://postgresql.ru.net/manual/history.html>. (дата обращения: 09.05.2017).
25. Мессенджеры в России: – Режим доступа: <http://blog.br-analytics.ru/messendzhery-vesna-2017/>. (дата обращения: 01.06.2017).
26. Метод анализа иерархий: – Режим доступа: <https://ru.wikipedia.org/wiki/Метод_анализа_иерархий>. (дата обращения: 15.05.2017).
27. Основы языка программирования Python за 10 минут: – Режим доступа: <https://habrahabr.ru/post/31180/>. (дата обращения: 13.05.2017).
28. Попов А. Л, Системы поддержки принятия решений: – Режим доступа: <http://elar.urfu.ru/bitstream/10995/1676/5/1335843_schoolbook.pdf>. (дата обращения: 02.03.2017).
29. Программирование на Python: Часть 1. Возможности языка и основы синтаксиса: – Режим доступа: <https://www.ibm.com/developerworks/ru/library/l-python_part_1/>. (дата обращения: 14.05.2017).
30. Розанов М. «BI в России: от простого к сложному»: – Режим доступа: <http://luxmsbi.com/index.php/ru/news/item/68-cnews-conference>. (дата обращения: 22.05.2017). [9]
31. Сервисы рекомендаций: как с их помощью увеличить продажи: – Режим доступа: <https://www.kom-dir.ru/article/51-servisy-rekomendatsiy>. (дата обращения: 27.05.2017).
32. Система поддержки принятия решений: – Режим доступа: http://www.tadviser.ru/index.phpСтатья:Система\_поддержки\_принятия\_решений#.D0.9F.D1.80.D0.B5.D0.B8.D0.BC.D1.83.D1.89.D0.B5.D1.81.D1.82.D0.B2.D0.B0. (дата обращения: 15.05.2017).
33. Системы поддержки принятия решений: – Режим доступа: http://bourabai.ru/tpoi/dss.htm#1. (дата обращения: 02.03.2017).
34. Создание систем поддержки принятия решений: – Режим доступа: <http://mirznanii.com/a/111739/sozdanie-sistem-podderzhki-prinyatiya-resheniy>. (дата обращения: 12.03.2017).
35. СУБД Postgres Pro: – Режим доступа: <https://reestr.minsvyaz.ru/reestr/65273/>. (дата обращения: 03.06.2017). [12]
36. Телеграм: – Режим доступа: <https://telegram.org>. (дата обращения: 11.04.2017).
37. Трахтенгерц Э. А. Компьютерная поддержка принятия решений: – Режим доступа: <http://mtas.ru/upload/library/trahtengerts1.pdf>. (дата обращения: 23.05.2017).
38. Ястребов В.С., Митихин В.Г., Михайлова И.И. Применение метода анализа иерархий в практике, психосоциальной реабилитации и в программе дестигматизации: – Режим доступа: http://www.psychiatry.ru/siteconst/userfiles/destigm.pdf. (дата обращения: 27.05.2017). [8]

Приложение А

Листинг программы.

import config #импортируем файл с токеном где лежит ключ

import telebot #сама библиотека

import pywapi #библиотека погоды

import postgresql #база данных

import random #библиотека случайных чисел

otvet=['лучше побудь дома', 'никуда не ходи']

bot=telebot.TeleBot(config.token) #токен наш в фалике конфиг имя token

@bot.message\_handler(commands=['start'])

def handle\_start(message):

user\_markup=telebot.types.ReplyKeyboardMarkup(True, False)

user\_markup.row('\u2601 Погода на сегодня', '/help \U0001f631')

user\_markup.row('\U0001f31d Погода на 3 дня \U0001f31a')

user\_markup.row('\U0001f61c Расскажи анекдот', 'Задать вопрос \U0001f4ac')

bot.send\_message(message.chat.id, 'Привет, '+message.chat.first\_name+' чем могу помочь?', reply\_markup=user\_markup)

@bot.message\_handler(commands=['help'])

def handle\_help(message):

bot.send\_message(message.chat.id, config.random\_message())

@bot.message\_handler(content\_types=['text'])

def handle\_anekdot(message):

if message.text=="\U0001f61c Расскажи анекдот":

bot.send\_message(message.chat.id, config.random\_anekdot())

if message.text=="\u2601 Погода на сегодня":

handle\_krasnoyarsk(message)

if message.text=="\U0001f31d Погода на 3 дня \U0001f31a":

handle\_krasnoyarskkk(message)

if message.text=="Задать вопрос \U0001f4ac":

handle\_dialog(message)

@bot.message\_handler(commands=['krasnoyarsk'])

def handle\_krasnoyarsk(message):

weather\_com\_result=pywapi.get\_weather\_from\_weather\_com('RSXX0267')

bb=weather\_com\_result['current\_conditions']['text']

if bb=='Fair':

bb='\u2600'

weatherReport=bb+"В Красноярске сейчас "+weather\_com\_result['current\_conditions']['temperature']+" °С "+"\n По ощущени-ям: "+weather\_com\_result['current\_conditions']['feels\_like']+" °С \n Влажность воз-духа "+weather\_com\_result['current\_conditions']['humidity']+' % \n поэтому '+random.choice(otvet)

aa=weather\_com\_result['current\_conditions']['temperature']

bot.send\_message(message.chat.id, weatherReport)

# bot.send\_message(message.chat.id, bb)

print(weather\_com\_result)

print('..............')

@bot.message\_handler(commands=['krasnoyarskkk'])

def handle\_krasnoyarskkk(message):

weather\_com\_result=pywapi.get\_weather\_from\_weather\_com('RSXX0267')

result=pywapi.get\_weather\_from\_weather\_com('RSXX0267')

a=''

b='Погода на ближайшие дни: \n'

for i in range(0,3):

c=result['forecasts'][i]['day\_of\_week']

d={

c=='Monday': 'Понедельник',

c=='Tuesday': 'Вторник',

c=='Wednesday': 'Среда',

c=='Thursday': 'Четверг',

c=='Friday': 'Пятница',

c=='Saturday': 'Суббота',

c=='Sunday': 'Воскресенье'}[1]

a="\n "+d+" "+"\n "+"Max: "+result['forecasts'][i]['high']+" °С \n "+"Min: "+result['forecasts'][i]['low']+" °С \n "

b=b+a

# weatherReport="Погода на 3 дня:"+weather\_com\_result['forecasts'][1]['date']+" "+weather\_com\_result['forecasts'][1]['day\_of\_week']

bot.send\_message(message.chat.id, b)

print(weather\_com\_result)

print(weather\_com\_result['forecasts'][1]['day\_of\_week'])

@bot.message\_handler(commands=['otvet'])

def handle\_otvet(message):

if message.text.lower() in config.dota:

bot.send\_message(message.chat.id, config.dota[message.text.lower()])

handle\_dialog(message)

@bot.message\_handler(commands=['dialog'])

def handle\_dialog(message):

user\_markup=telebot.types.ReplyKeyboardMarkup(True, False)

user\_markup.row('Что там с погодой?', 'Что мне одеть сегодня?')

user\_markup.row('Как там африканские дети?', 'Вернуться в меню')

msg=bot.send\_message(message.chat.id, "Что ты хотел спросить?", re-ply\_markup=user\_markup)

Приложение Б

В соответствии с рисунком 18 представлен скриншот разработанного бота для мессенджера telegram.

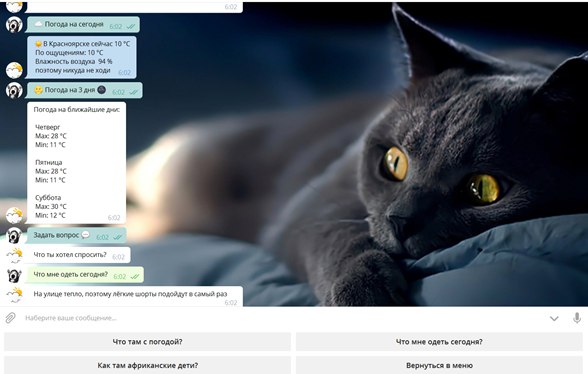


Рисунок 18 - Скриншот работы готового приложения