Московский государственный
университет имени М.В. Ломоносова

Факультет космических исследований

Магистерская программа
"Методы и технологии дистанционного
зондирования Земли"

Выпускная квалификационная работа

Разработка системы ведения архива и контроля обработки исходных данных миссий наблюдения Марса

Шадрина Анастасия Владимировна
Научный руководитель:
к.т.н., Бурцев Михаил Александрович

Москва, 2020

Содержание

ВВЕДЕНИЕ

Начиная с первой миссии Mariner 9 по исследованию Марса, на спутники устанавливалось различное оборудование, с помощью которого можно было получить изображения поверхности этой планеты. С течением времени оборудование усложнялось, увеличивался объем получаемых данных.

Сегодня для работы с такими данными создаются специальные информационные системы, одной из которых является ARES-Mars. Отличительной особенностью этого сервиса являются инструменты, с помощью которых возможно проводить обработку и анализ данных на основе имеющихся архивов, что позволяет пользователю не тратить дополнительно собственные ресурсы на скачивание и подготовку данных.

Предлагаемый подход даёт возможность проведения онлайн-обработки любых данных архивов за счет ресурсов сервиса. Кроме этого, интерфейс системы позволяет работать с данными миссий по изучению Марса посредством широко распространённых веб-браузеров, без использования дополнительного ПО для обработки данных. [1]

Система создается на базе разработанных в ИКИ РАН технологий UNISAT[2] и GEOSMIS[3], что позволит применить существующие наработки и подходы для работы с пространственными данными дистанционного зондирования Земли и их анализа к исследованиям Марса.

1 ПОСТАНОВКА ЗАДАЧИ

Основной задачей настоящей работы является создание программного обеспечения, позволяющего вести архив и контроль обработки данных, которые были получены в ходе миссий наблюдения Марса.

В рамках данной работы происходит обращение к планетарной информационной системе NASA (PDS), которая распространяет подобные цифровые данные, архивы находятся в режиме онлайн и доступны для бесплатного скачивания.

Разработанное приложение должно выполнять следующие задачи:

1. Посредством HTTP-запроса к внешним архивам данных получать актуальную информацию о добавлении новых или обновлении существующих данных

2. Выполнять загрузку открытых исходных данных с серверов их расположения

3. Подготовливать данные для дальнейшей загрузки в базу данных

4. Загружать подготовленные данные в базу данных в автоматическом режиме

5. Вести журнал событий

6. Добавлять информацию о дальнейшей обработке исходных данных

Для реализации данной задачи также необходимо определить структуру базы данных, которая будет хранить в себе всю информацию о полученных данных и результатах их обработки.

В конечном итоге, полученный результат – программное обеспечение в связке с базой данных – должен быть интегрирован в систему ARES-Mars, расширяя её возможности.

Общая схема работы с данными представлена на Рис.1, отдельным контуром на ней выделены этапы, которые автоматически выполняются с по-

мощью реализованного в ходе выполнения настоящей работы программного обеспечения:

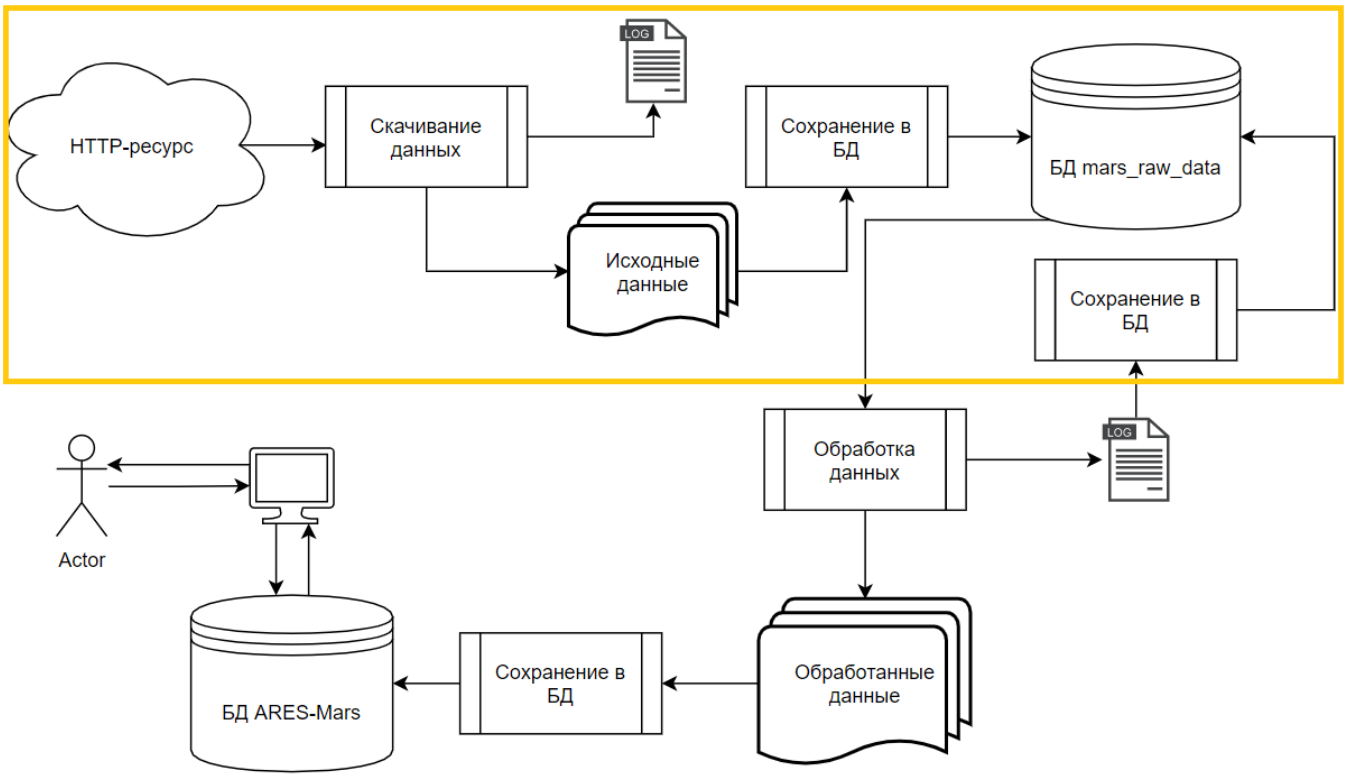


Рис. 1: Модель работы с данными

2 ОБЗОР ИНФОРМАЦИОННОЙ СИСТЕМЫ ARES-Mars

Для обеспечения совместной работы с данными приборов Trace Gas Orbiter (TGO) миссии Exomars-2016 в ИКИ РАН на основе технологий UNISAT и GEOSMIS, ранее применяемых для анализа данных дистанционного зондирования Земли, была создана специализированная веб-ГИС, получившая название ARES-Mars – Analysis, Research and Exploration Service.

Система ARES-Mars позволяет получить совместный доступ как данным, поступающим с TGO, так и к данным предыдущих миссий по исследованию Марса. Такой подход не только позволяет достичь поставленной цели, касающейся совместной работы с разнородными данными текущих миссий, но и позволяет решить эту задачу для исторических данных, по отдельности накопленных для разных миссий. Сегодня в дополнение к информации, поступающей с космического аппарата TGO, в систему интегрируются данные с оптических камер, установленных на КА Mars Express и ведутся работы по интеграции оптических данных, полученных с космического аппарата Mars Reconnaissance Orbiter [4].

Главной особенностью, отличающей сегодня миссии по наблюдению Марса от миссий по наблюдению Земли, является тот факт, что каждый новый прибор создается под свои уникальные задачи, а устоявшийся формат распространяемых данных подходит в первую очередь для научных исследований.

Система ARES-Mars объединяет в себе инструменты, которые позволяют проводить распределенную обработку и совместный анализ данных текущих и исторических миссий по изучению Марса. Глубокая интеграция с постоянно пополняемыми архивами данных, использование ресурсов системы, возможность выбора данных по заданным критериям и работа на основе широко распространенных веб-браузеров позволяет отказаться от традиционных подходов к работе с данными дистанционных исследований, требующих проведения исследователями большого количества процедур обработки данных на своих рабочих мощностях и перекачки больших объемов данных, а так-

же отказаться от использования дополнительного программного обеспечения для обработки данных.

Система включает в себя три основных компонента:

1. Блок ведения архивов;
2. Блок обработки данных;
3. Блок обеспечения работы с данными.

Созданный аналитический интерфейс ARES-Mars предназначен для работы с данными в формате, адаптированном для быстрой визуализации и онлайн-обработки (например, формат GeoTIFF для растровых данных). Все входные данные конвертируются в этот формат, после чего отправляются в подсистему архивации сервиса и становятся доступны для анализа в интерфейсе.

Технологически система архивации построена на основе технологии UNISAT, разработанной в ИКИ РАН для работы с разнородными данными наблюдения Земли. Ее общие принципы и подходы оказались достаточно универсальными, что позволило применить ее к данным наблюдения других планет с минимальной адаптацией.

Можно выделить следующие ключевые характеристики технологии UNISAT [5]:

1. Использование унифицированной структуры БД, позволяющей оптимально хранить разнородные спутниковые данные и результаты их обработки;
2. Наличие единой справочной БД, которая содержит в себе информацию о центрах приема, спутниках, приборах, продуктах, включая «виртуальные» продукты и правила их построения;
3. Функционал, имеющийся в специализированных системах ведения архивов спутниковых данных, позволяет обеспечивать доступ к спутниковым данным различных типов;

4. Поддержка единого для всех типов спутниковых данных механизма предоставления гибкого доступа к «виртуальным» продуктам;

5. Для анализа данных и проведения обработки в режиме реального времени поддерживается предоставление расширенных метаданных;

Внедрение данной технологии, обеспечивающей эффективную работу со сверхбольшими распределенными архивами спутниковых данных, позволило реализовать систему работы с планетными данными, позволяющую проводить комплексный анализ разнородной информации с использованием единого инструментария.

Основным инструментом работы с данными в системе ARES-Mars являются специализированные веб-картографические интерфейсы, созданные с использованием технологии GEOSMIS, разработанной в ИКИ РАН. Эти интерфейсы позволяют пользователю работать с разнородными данными, позволяя задавать произвольные временные и пространственные ограничения. В дополнение к возможности выбора и просмотра данных, картографические интерфейсы предоставляют возможность обрабатывать и анализировать данные в режиме онлайн. Кроме того, пользователь может использовать механизмы анализа различных качественных и количественных показателей наблюдаемых территорий, таких как динамика различных показателей для заданных территорий или пространственные профили по заданным линиям разреза.

3 ОБЗОР ДЕЙСТВУЮЩИХ МИССИЙ ПО ИЗУЧЕНИЮ МАРСА

В рамках настоящей работы были рассмотрены следующие действующие миссии по изучению Марса:

1. Mars Global Surveyor. Дата запуска: 7 ноября 1996 года. Орбитальный аппарат исследовал поверхность Марса и отслеживал ее глобальные погодные условия, работая дольше, чем любой предыдущий космический корабль, отправленный на Марс в то время. Mars Global Surveyor вернул более 240 000 изображений и 206 миллионов измерений спектрометра. Его лазерный альтиметр сработал 671 миллион раз, что позволило создать почти глобальную топографическую карту планеты.

2. Mars Odyssey. Дата запуска: 7 апреля 2001 года. Главная задача, стоящая перед аппаратом, заключается в изучении геологического строения планеты и поиске минералов. Установленный на аппарате лучевой гамма-спектрометр передал данные, свидетельствующие о крупных запасах воды на Марсе. Анализ данных о температуре позволил ученым провести различие между твердой породой и различными рыхлыми поверхностными материалами. С помощью системы камер планета исследуется как в видимом, так и в инфракрасном диапазонах, что уже позволило определить минералы в скалах и почвах, а также составить глобальную карту Марса в достаточно высоком разрешении. «Одиссей» является самым долгодействующим космическим аппаратом из отправленных на Марс, работающий в том числе и в качестве ретранслятора для передачи информации с марсоходов «Спирит» и «Оппортьюнити».

3. Mars Express. Дата запуска: 2 июня 2003 г. Орбитальный аппарат Mars Express передает цветные изображения и другие данные с января 2004 г. Более 95% поверхности планеты было получено с помощью камеры высокого разрешения, причем две трети отображены с разрешением выше 20 м на пиксель. Его спектрометр обнаружил залежи глинистых

минералов, указывающих на наличие влажной среды с низкой кислотностью. Один из установленных на аппарате радаров показал наличие подлёдного озера, которое стало первым известным постоянным водоёмом на Марсе[6]. Кроме того, во время миссии в атмосфере Марса был обнаружен метан, что вероятно может свидетельствовать о наличии тектонической активности. Серия полетов, близких к спутнику Марса Фобосу, дает возможность изучить состав поверности и происхождение спутника.

4. Mars Reconnaissance Orbiter. Дата запуска: 12 августа 2005 г. Главная задача миссии заключается в анализе рельефа, стратиграфии, изучении минералов и льда на Марсе, а также исследовании погоды и поверхности Марса для поиска возможных мест посадки. Орбитальный аппарат отправил назад более чем в три раза больше данных, чем все другие космические миссии, которые отправились за пределы Луны. NASA получили более 190 терабайт данных – в том числе более 70 000 изображений – от шести научных приборов в течение первых восьми лет полета миссии на Марс.

3.1 Mars Express

Для получения тестовых исходных данных была выбрана миссия Mars Express ввиду того, что архив данных этой миссии имеет наиболее актуальную информацию. Оборудование, установленное на спутнике[7]:

- Анализатор заряженных частиц — ASPERA
- Цветная стереокамера высокого/сверхвысокого разрешения — HRSC
- Инфракрасный спектрометр — OMEGA
- Атмосферный спектрометр Фурье — PFS
- Радиопередатчик — RSE
- Ультрафиолетовый атмосферный спектрометр — SPICAM

- Подповерхностный радар / высотомер — SSRA

Для дальнейшей работыиспользуются данные, полученные с помощью HRSC – стереоскопической камеры, которая фотографирует марсианскую поверхность, чтобы показать детали с разрешением до 2 метров. Изображения используются для создания геологической карты, показывающей местонахождение различных минералов и типов пород.

Основным источником данных HRSC является архив ESA Planetary Science[8]. В качестве исходных данных предлагается выбрать комплекты заданного формата:

- *.lbl
- *.img
- *.jp2

4 РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Данная работа является продолжением курсовой работы[9], в рамках которой были рассмотрены, реализованы и протестированы некоторые из распространенных методов для начальной работы с данными. Полученные результаты были расширены и учтены при написании настоящей работы.

Для решения поставленных задач, в частности, создания программного обеспечения, позволяющего загружать исходные данные, производить их обработку, генерировать входной файл для дальнейшей работы с СУБД MySQL и вести логирование выполненных операций, был выбран язык программирования Python. Структура кода, написанного на этом языке, понятна, есть возможность интеграции с другими языками программирования, язык является кроссплатформенным и имеет достаточное количество библиотек для работы с данными, которые используются в рамках настоящей работы.

4.1 Работа с данными

Первым делом необходимо получить доступ к внешним архивам данных. Для этой цели необходимо выполнить GET-запрос к HTTP-серверу, возвращающий объект Response. На основе полученной информации делается вывод о структуре директории с данными, извлекается информация о наличии новых данных или обновлении уже существующих. Следующим этапом производится предварительная обработка данных и их непосредственная загрузка во внутреннее хранилище. Рассмотрим подробнее каждый из этапов.

Для языка программирования Python существует множество библиотек, методами которых можно осуществить GET-запрос. В настоящей работе используется фреймворк aiohttp[10], работающий на основе библиотеки асинхронных вызовов asyncio[11], поскольку он обладает важным преимуществом – он позволяет отправлять запросы по нескольким соединениям параллельно, избегая проблемы блокировки в режиме ожидания ответа от HTTP-сервера. Это важно, так как большие объемы данных, которые необходимо загрузить,

порождают множественные запросы к серверу. Синхронное или последовательное выполнение этих запросов потребует немалых ресурсов памяти и времени.

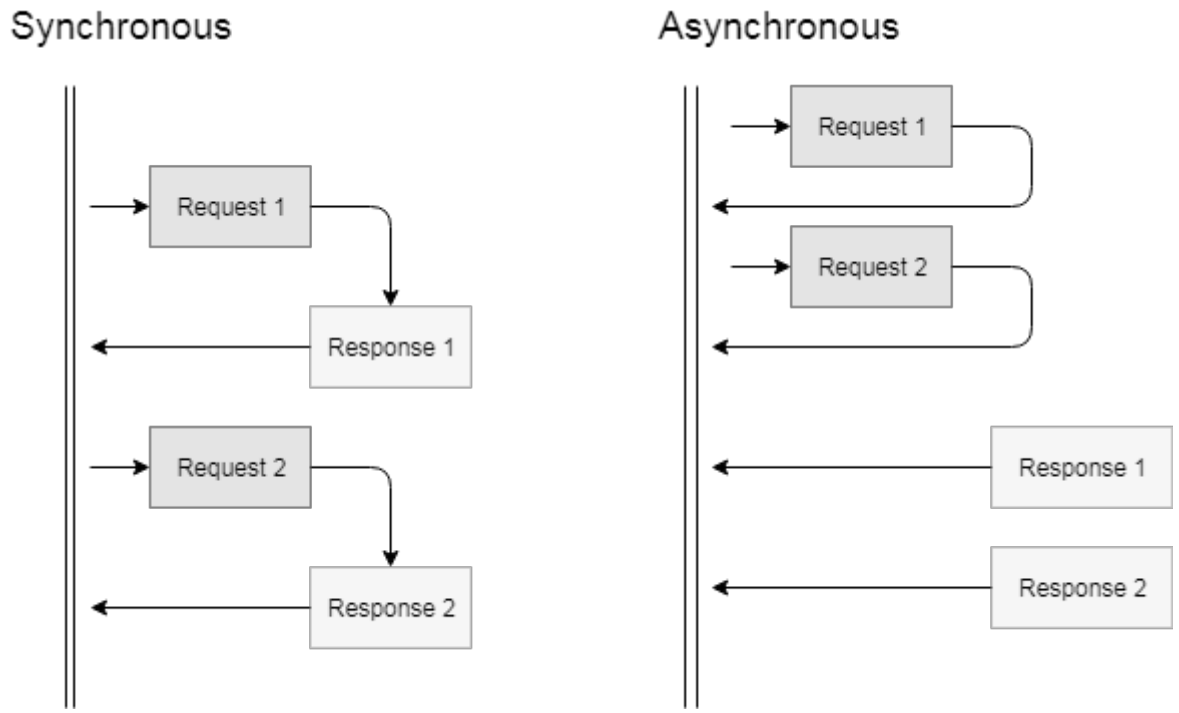


Рис. 2: Различие синхронных и асинхронных HTTP-запросов

4.1.1 Асинхронные методы обработки данных

Асинхронное программирование позволяет писать параллельный код, который выполняется в одном потоке. Первое преимущество по сравнению с несколькими потоками заключается в том, что можно программно указать, где планировщик будет переключаться с одной задачи на другую, а это означает, что обмен данными между задачами безопаснее и проще.

Еще одним преимуществом асинхронного программирования является использование памяти. Каждый раз, когда создается новый поток, часть памяти используется для переключения контекста. Если мы используем асинхронное программирование, это не проблема, так как код выполняется в одном потоке.

Asyncio имеет 3 основных компонента:

- сопрограммы (coroutines) – результат асинхронной функции

- цикл событий (event loop) – объект, который выполняет наш асинхронный код и решает, как переключаться между асинхронными функциями
- фьючерс (future) – задание для выполнения сопрограммы из цикла событий

Для создания клиент-серверной сессии и выполнения HTTP-запросов используется метод aiohttp.ClientSession, который может принимать множество параметров, но для наших задач все они являются необязательными.

На первом этапе проанализируем страницу, полученную в виде HTML в результате GET-запроса. Возможности языка программирования Python позволяют сделать это несколькими возможными способами (регулярные выражения, библиотека BeautifulSoup, библиотека lxml, фреймворк scrapy), которые по своей сути отличаются только удобством использования. Измерение величины времени обработки веб-страницы этими методами не дало существенных различий в результатах. В рамках настоящей работы используется обработка с использованием регулярных выражений. В Python они реализованы в стандартном модуле re[12]. Для парсинга достаточно использования метода re.findall(pattern, string), который возвращает список кортежей, группирующих все вхождения, соответствующие заданном шаблону. Здесь стоит отметить, что полученная веб-страница имеет определенную структуру[13] вида:

Дата	Время	Формат времени	Тип объекта	Ссылка на объект

С помощью регулярного выражения выделяем дату последнего обновления объекта (в данном случае директории) и ссылку на объект. По умолчанию, у нас есть лог-файл, последняя строка которого содержит в себе дату последней проверки внешних архивов данных:

```
2020-05-06 04:05:55.186186 hj071_0000_nd3.img
2020-05-06 04:06:25.952728 hj001_0000_nd3.img
2020-05-06 04:06:37.519816 hi984_0000_nd3.img
2020-05-06 04:06:54.654657 hi995_0000_nd3.img
2020-05-06 04:08:32.113429 hi976_0000_nd3.img
END: 2020-05-06 04:08:32.782383
LAST UPDATE: 2020-05-06
```

Рис. 3: Пример записи в лог-файле

На основе полученной информации производится поиск директорий, которые были обновлены или добавлены позже указанной в лог-файле даты:

```
START: 2020-05-06 03:06:23.985001
READ FOLDERS: 4575
NEW FOLDERS: 37
NEW FILES: 966
SIZE OF FILES: 45.38 GB
```

Рис. 4: Пример записи в лог-файле

После получения информации об обновленных директориях и записи ее в лог-файл создается следующий набор задач, который состоит из подпрограмм, выполняющих поиск информации о файлах в каждой конкретной директории, и запускается методом run_until_complete. В ходе выполнения подпрограммы создается очередная клиент-серверная сессия, где выполняется GET-запрос. Анализируя полученные данные, выделяем необходимую для дальнейшей работы информацию:

- имя файла
- тип файла
- размер файла (в байтах)
- дата обновления файла
- орбита, на которой был сделан снимок
- дата обновления директории

Вся информация объединяется в объект класса File (file_name, file_type, file_size, file_update, orbit_id, orbit_update). Эта операция выполняется последовательно для каждого найденного в директории файла и асинхронно для всех директорий. Все объекты помещаются в специальный список.

По завершении этой операции и записи в лог-файл информации об общем количестве файлов и суммарной размерности, но до их скачивания необходимо разбить подпрограммы скачивания на так называемые подочереди. Это необходимое условие, поскольку даже несмотря на асинхронность выполнения задач без разбиения нагрузка на сервер создастся колоссальная, что приведет к преждевременному прерыванию процесса. Эмпирическим путем – на основе анализа информации о том, как часто выполняется обновление внешних архивов и сколько в среднем добавлено файлов – было выбрано разбиение по 200 задач, это решение вкупе с использованием семафоров (метод asyncio.Semaphore) оказалось оптимальным для снижения нагрузки.

Во время последовательного запуска каждой из частей общего объема задач, выполняются следующие действия:

1. GET-запрос, в качестве параметра передается ссылка на файл и снятие ограничения на ожидание ответа. Последнее обусловлено большим размером некоторых файлов.

2. Чтение полученного Responce-объекта

3. Создание файла, в который объект будет записан

4. Сохранение объекта в файл

5. Создание соответствующей записи в лог-файл

6. Генерация записи в файл, который в дальнейшем будет использован для записи всех скачанных файлов в базу данных

После завершения работы программы переходим к этапу работы с базой данных.

4.2 Работа с базой данных

База данных (БД) — это организованная структура, предназначенная для хранения, изменения и обработки взаимосвязанной информации, преимущественно больших объемов.

Система управления базами данных (СУБД) — это комплекс программных средств, необходимых для создания структуры новой базы, ее наполнения, редактирования содержимого и отображения информации.

Основные функции СУБД:

- управление данными во внешней памяти (на дисках);
- управление данными в оперативной памяти с использованием дискового кэша;
- журнализация изменений, резервное копирование и восстановление базы данных после сбоев;
- поддержка языков БД (язык определения данных, язык манипулирования данными).

Наиболее распространенными реляционными СУБД являются MySQL, PostgreSQL, Microsoft SQL Server. Для задачи системы ведения архивов и контроля обработки данных используется система управления базами данных MySQL[14]. Она относится к классу клиент-серверных СУБД. Клиент-серверная СУБД располагается на сервере вместе с БД и осуществляет доступ к БД непосредственно, в монопольном режиме. Все клиентские запросы на обработку данных обрабатываются клиент-серверной СУБД централизованно[15]. Недостаток клиент-серверных СУБД состоит в повышенных требованиях к серверу. Достоинства:

- потенциально более низкая загрузка локальной сети;
- удобство централизованного управления;
- удобство обеспечения таких важных характеристик, как высокая надёжность, высокая доступность и высокая безопасность.

Из преимуществ СУБД MySQL также стоит отметить простоту использования, гибкость, масштабируемость.

С учетом специфики данных, была сформирована следующая структура БД:

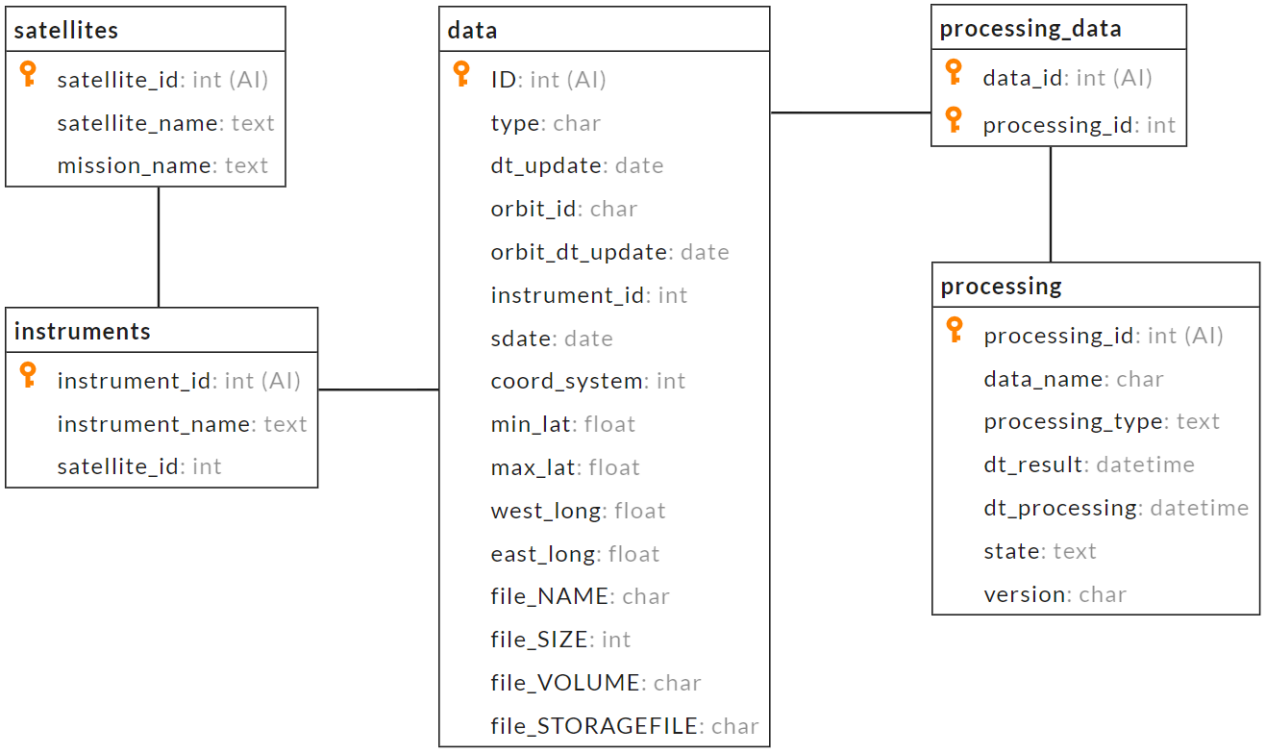


Рис. 5: Структура базы данных mars_raw_data

Для создания таблицы data и наполнения ее данными используется пакет утилит FDB (File Data Base)[16]. В совокупности с разработанной структурой таблиц данный пакет предназначен для построения систем администрирования баз данных, ориентированных на хранение файлов. Фактически он является своего рода надстройкой сервера реляционных баз данных, позволяющей хранить в БД объект типа файл и производить над ним необходимые операции.

За выполнение каждой из базовых операций с FDB таблицей, таких как создание таблицы, добавление и удаление записей отвечает соответствующая утилита.

- CREATE – создание новой FDB таблицы
- ADD – добавление новых записей или обновление существующих

- REMOVE – удаление записей
- DETACH – перенос файлов на автономный носитель, удаление файлов
- ATTACH – импортирование файлов с автономного носителя

В структуре таблиц объект "файл данных"реализуется четырьмя полями с именами, содержащими одинаковый префикс, обозначающий имя объекта ”файл” и постфиксами, по которым утилиты могут распознать его поля-компоненты. Поля содержат информацию о первоначальном имени файла, его размере, томе архива и имени под которым файл хранится в базе данных.

FDB утилиты имеют двухуровневую систему ведения лог файлов. Лог файл верхнего уровня содержит протокол выполнения утилит. В лог файл нижнего уровня заносится детальная информация о производимых файловых операциях и SQL командах, модифицирующих таблицу БД.

Таким образом, в соответствии с ранее определенной структурой, посредством утилиты CREATE в БД mars_raw_data создана таблица data. Одним из параметров является входной SQL-файл, определяющий структуру данной таблицы:

```
create table data
(
    ID int auto_increment primary key,
    type char(3) NOT NULL,
    dt_update date NOT NULL,
    orbit_id char(4) NOT NULL,
    orbit_dt_update date NOT NULL,
    instrument_id int NOT NULL,
    sdate date NOT NULL,
    coord_system char(14) NOT NULL,
    min_lat float(7,4) NOT NULL,
    max_lat float(7,4) NOT NULL,
    west_long float(7,4) NOT NULL,
    east_long float(7,4) NOT NULL,
    file_NAME char(50) default NULL,
    file_SIZE int default NULL,
    file_VOLUME char(30) default NULL,
    file_STORAGEFILE char(30) default NULL
);
```

Рис. 6: Создание таблицы БД data

Следующий этап – внесение в БД скачанных ранее данных. Для осуществления этой операции вернемся к файлу, который был сгенерирован программой в ходе загрузки данных. Формат каждого поля записи соответствует структуре полей таблицы БД:

```
jp2|2020-04-29|j236|2020-05-01|1|2019-02-04|PLANETOGRAPHIC|-54.4073|-39.672|307.678|310.821|j236_0001_nd3.jp2|3610778|NULL|NULL
jp2|2020-04-29|j629|2020-05-01|1|2019-02-04|PLANETOGRAPHIC|-54.3973|-39.6915|307.61|310.777|j629_0003_nd3.jp2|546052|NULL|NULL
img|2020-04-29|j489|2020-05-01|1|2019-02-04|PLANETOGRAPHIC|-54.3301|-39.6734|307.476|310.676|j489_0005_nd3.img|2354162|NULL|NULL
lbl|2020-04-29|j995|2020-05-01|1|2019-02-04|PLANETOGRAPHIC|-54.3833|-39.6749|307.662|310.794|j995_0001_nd3.lbl|4349|NULL|NULL
jp2|2020-04-29|j479|2020-05-01|1|2019-02-04|PLANETOGRAPHIC|-54.328|-39.6602|307.515|310.706|j479_0002_nd3.jp2|1064159|NULL|NULL
jp2|2020-04-29|j208|2020-05-01|1|2019-02-04|PLANETOGRAPHIC|-54.4543|-39.6807|307.787|310.871|j208_0008_nd3.jp2|3643568|NULL|NULL
lbl|2020-04-29|j198|2020-05-01|1|2019-02-04|PLANETOGRAPHIC|-54.4691|-39.6763|307.863|310.889|j198_0004_nd3.lbl|4056|NULL|NULL
lbl|2020-04-29|j299|2020-05-01|1|2019-02-04|PLANETOGRAPHIC|-54.3212|-39.6848|307.427|310.652|j299_0002_nd3.lbl|1394|NULL|NULL
jp2|2020-04-29|j713|2020-05-01|1|2019-02-04|PLANETOGRAPHIC|-54.4985|-39.6501|307.847|310.917|j713_0001_nd3.jp2|2619288|NULL|NULL
```

Рис. 7: Пример записей для занесений файлов в БД

Утилита ADD для добавления файлов в базу данных имеет следующий формат: add [options] <db> <table> <source dir> < {add_file} > {id_file}, где

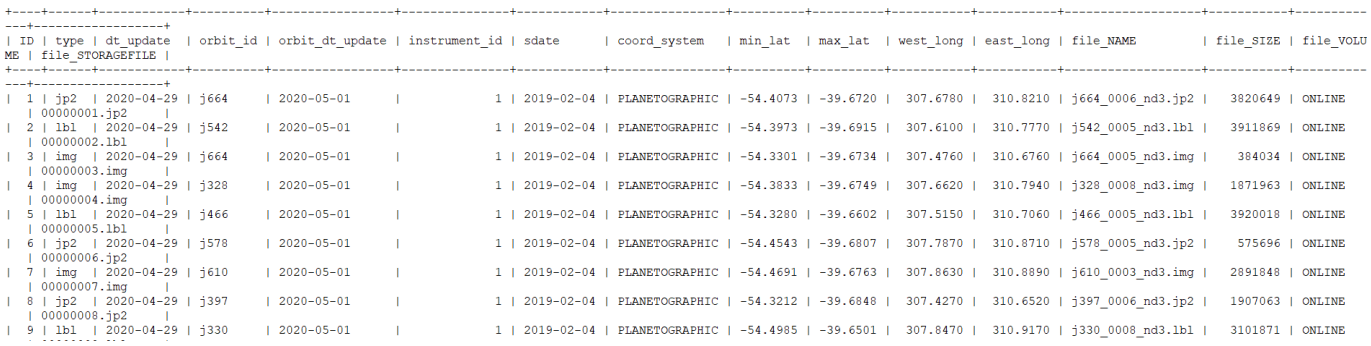
- <db> – название базы данных (в нашем случае mars_raw_data)
- <table> – название таблицы (data)
- <source dir> – директория, в которой хранятся исходные данные для загрузки
- {add_file} – документ, каждая строка которой содержит запись о добавляемом файле
- {id_file} – документ, в который будут записаны ID всех добавленных файлов

Утилита также имеет несколько опций, например, можно указать параметры записи или указать конфигурационный файл (по умолчанию "./fdb.conf"). Последний должен содержать в себе следующие параметры:

- FILE_STORAGE_ROOT_DIR – корневая директория локального хранилища файлов
- NEW_STORAGE_STYLE – схема хранения
- MYSQL_HOST – адрес хоста на котором находится БД сервер

- MYSQL_USER – пользователь MySQL
- MYSQL_PASSWORD – пароль пользователя MySQL

Добавленные в базу данных записи о файлах выглядят следующим образом:



ID	type	dt_update	orbit_id	orbit_dt_update	instrument_id	sdate	coord_system	min_lat	max_lat	west_long	east_long	file_NAME	file_SIZE	file_VOLUME	file_STORAGEFILE
1	jp2	2020-04-29	j664	2020-05-01	1	2019-02-04	PLANETOGRAPHIC	-54.4073	-39.6720	307.6780	310.8210	j664_0006_nd3.jp2	3820649	ONLINE	00000001.jp2
2	lbl	2020-04-29	j542	2020-05-01	1	2019-02-04	PLANETOGRAPHIC	-54.3973	-39.6915	307.6100	310.7770	j542_0005_nd3.lbl	3911869	ONLINE	00000002.lbl
3	img	2020-04-29	j664	2020-05-01	1	2019-02-04	PLANETOGRAPHIC	-54.3301	-39.6734	307.4760	310.6760	j664_0005_nd3.img	384034	ONLINE	00000003.img
4	img	2020-04-29	j328	2020-05-01	1	2019-02-04	PLANETOGRAPHIC	-54.3833	-39.6749	307.6620	310.7940	j328_0008_nd3.img	1871963	ONLINE	00000004.img
5	lbl	2020-04-29	j466	2020-05-01	1	2019-02-04	PLANETOGRAPHIC	-54.3280	-39.6602	307.5150	310.7060	j466_0005_nd3.lbl	3920018	ONLINE	00000005.lbl
6	jp2	2020-04-29	j578	2020-05-01	1	2019-02-04	PLANETOGRAPHIC	-54.4543	-39.6807	307.7870	310.8710	j578_0005_nd3.jp2	575696	ONLINE	00000006.jp2
7	img	2020-04-29	j610	2020-05-01	1	2019-02-04	PLANETOGRAPHIC	-54.4691	-39.6763	307.8630	310.8890	j610_0003_nd3.img	2891848	ONLINE	00000007.img
8	jp2	2020-04-29	j397	2020-05-01	1	2019-02-04	PLANETOGRAPHIC	-54.3212	-39.6848	307.4270	310.6520	j397_0006_nd3.jp2	1907063	ONLINE	00000008.jp2
9	lbl	2020-04-29	j330	2020-05-01	1	2019-02-04	PLANETOGRAPHIC	-54.4985	-39.6501	307.8470	310.9170	j330_0008_nd3.lbl	3101871	ONLINE	

Рис. 8: Пример записей в БД mars_raw_data

Последним этапом является организация контроля версий. Для решения этой задачи был написан скрипт на языке Python, одним из входных параметров которого является лог-файл. Этот файл должен генерироваться во время обработки каких-либо исходных данных, содержащихся в таблице. Структура этого файла может выглядеть следующим образом:

- <имя исходного файла> – имя файла, под которым он хранится после его загрузки в таблицу через FDB-утилиту
- <имя обработанного файла> – имя файла, под которым он будет сохранен после обработки
- <тип> – тип операции обработки, которая совершалась над исходным файлом
- <статус операции> – статус, с которым могла завершиться операция
- <дата и время> – время выполнения соответствующей операции

```
2.jpg t2.jpg train successful 2020-05-06 16:20:52.638744
3.jpg t3.jpg train successful 2020-05-07 16:20:52.638744
4.jpg t4.jpg train successful 2020-05-08 16:20:52.638744
1.jpg t1.jpg train successful 2020-05-09 16:20:52.638744
```

Рис. 9: Пример лог-файла

При каждом новом прочтении лог-файла генерируется документ, который уже непосредственно содержит информацию о каждой операции в виде, удобном для добавления в таблицу processing:

- <data_name> – исходное имя файла
- <processing_type> – тип операции
- <dt_result> – дата и время занесения создания записи для занесения в таблицу
- <dt_processing> – дата и время выполнения операции
- <state> – статус операции
- <version> – имя обработанного файла

```
USE mars_raw_data;
INSERT processing(data_name, processing_type, dt_result, dt_processing, state, version) VALUES
('2.jpg', 'train', '2020-05-07 01:33:45', '2020-05-06 16:20:52', 'successful', 't2.jpg'),
('3.jpg', 'train', '2020-05-07 01:33:45', '2020-05-07 16:20:52', 'successful', 't3.jpg'),
('4.jpg', 'train', '2020-05-07 01:33:45', '2020-05-08 16:20:52', 'successful', 't4.jpg'),
('1.jpg', 'train', '2020-05-07 01:33:45', '2020-05-09 16:20:5', 'successful', 't1.jpg');
```

Рис. 10: Пример файла с операцией добавления в таблицу processing

В результате работы этой программы в таблицу были внесены записи следующего вида:

```
+---------------+-----------+-----------------+---------------------+---------------------+------------+---------+
| processing_id | data_name | processing_type | dt_result           | dt_processing       | state      | version |
+---------------+-----------+-----------------+---------------------+---------------------+------------+---------+
|             1 | 2.jpg     | train           | 2020-05-07 01:30:53 | 2020-05-06 16:20:52 | successful | t2.jpg  |
|             2 | 3.jpg     | train           | 2020-05-07 01:30:53 | 2020-05-07 16:20:52 | successful | t3.jpg  |
|             3 | 4.jpg     | train           | 2020-05-07 01:30:53 | 2020-05-08 16:20:52 | successful | t4.jpg  |
|             4 | 1.jpg     | train           | 2020-05-07 01:30:53 | 2020-05-09 16:20:05 | successful | t1.jpg  |
```

Рис. 11: Пример записей в таблице processing

Для удобства доступа к записям из таблицы processing, создана таблица-связка processing_data, которая содержит в себе соответствия ID исходного файла и ID записи об операции обработки.

Конечный этап – автоматизация всего процесса работы с исходными данными – выполнен как реализация алгоритма выделенной на Рис.1 части схемы в виде сценариев для выполнения в UNIX-среде. Запуск срипта, содержащего данные сценарии, организует последовательное выполнение всех частей программного обеспечения.

5 РЕЗУЛЬТАТЫ РАБОТЫ

Таким образом, в ходе выполнения работы была решена задача создания инструмента, полностью автоматизирующего процесс выполнения пакета скриптов для работы с исходными данными от загрузки до контроля последующей их обработки.

В рамках этого инструмента в частности были выполнены следующие подзадачи:

1. Организован доступ к файловым архивам данных миссии Mars Express, полученных с аппарата HRSC
2. Реализовано программное обеспечение, осуществляющее проверку обновлений, загрузку и первичную обработку файловых данных
3. Создана БД для хранения исходных данных с удобной для поиска по различным критериям архитектурой
4. Реализована программа для загрузки файлов в базу данных
5. Организовано логирование процесса на всех этапах первичной обработки данных
6. Реализовано программное обеспечение, позволяющее сохранять информацию о последующей обработке исходных данных.

Полученные результаты в перспективе могут использоваться для следующих целей [17]:

1. Обработка исходных данных с последующей фиксацией результатов в таблицах созданной базы данных
2. Интеграция базы данных в систему ARES-Mars для дальнейшей работы с загруженными данными

Список литературы

[1] Система ARES для работы с данными наблюдения Марса / Балашов И.В., Бурцев М.А., Сычугов И.Г. [и др.] // Материалы Пятнадцатой Всероссийской открытой конференции "Современные проблемы дистанционного зондирования Земли из космоса". 2017.

[2] Создание унифицированной системы ведения архивов спутниковых данных, предназначенной для построения современных систем дистанционного мониторинга / Прошин А.А., Лупян Е.А., Балашов И.В. [и др.] // Современные проблемы дистанционного зондирования Земли из космоса. Т. 13. № 3. С. 9-27. 2016.

[3] Создание интерфейсов для работы с данными современных систем дистанционного мониторинга (система GEOSMIS) / Толпин В.А., Балашов И.В., Ефремов В.Ю. [и др.] // Современные проблемы дистанционного зондирования Земли из космоса. Т.8. № 3. С. 93-108. 2011.

[4] 2020. URL: http://www.smislab.ru/default.aspx?page=741.

[5] 2020. URL: http://smiswww.iki.rssi.ru/default.aspx?page=546.

[6] 2018. URL: https://edition.cnn.com/2018/07/25/world/mars-subsurface-water-lake-evidence/index.html.

[7] 2020. URL: pds-geosciences.wustl.edu/missions/mars.express/default.html.

[8] 2020. URL: psa.esac.esa.int/pub/mirror/MARS-EXPRESS/HRSC/.

[9] А.В. Шадрина. Интеграция данных действующих миссий наблюдения Марса в ИС ARES-Mars. 2019.

[10] 2020. URL: https://docs.aiohttp.org.

[11] 2020. URL: https://docs.python.org/3/library/asyncio.html.

[12] 2020. URL: https://docs.python.org/3/library/re.html.

[13] 2020. URL: https://pds-geosciences.wustl.edu/mex/mex-m-hrsc-5-refdr-mapprojected-v3/mexhrs_1003/aareadme.txt.

[14] 2020. URL: https://dev.mysql.com/doc/.

[15] 2020. URL: https://ru.wikipedia.org/wiki/Система_управления_базами_данных.

[16] Описание пакета FDB. 2000.

[17] Интеграция данных различных миссий по иссследованию Марса в ИС ARES-Mars / Бурцев М.А., Шадрина А.В., Атрохов А.А. [и др.] // Материалы Семнадцатой Всероссийской Открытой конференции "Современные проблемы дистанционного зондирования Земли из космоса". С. 358. 2019.