УТВЕРЖДЕН
ДССЛ.00115-01 31 01 - ЛУ

СПЕЦИАЛЬНОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ

«3i Big Data Analytics Processing Platform»

Описание применения

ДССЛ.00115-01 31 01

Листов 64

Литера $О_1$

2015

АННОТАЦИЯ

Настоящий документ предназначен для ознакомления со специальным программным обеспечением (СПО) «3i Big Data Analytics Processing Platform» и содержит описание интерфейса программирования (API) для программистов, обеспечивающих использование Big Data Analytics Processing Platform в качестве модуля, встраиваемого в другое программное обеспечение.

В разделе 1 приводятся сведения о назначении 3i Big Data Analytics Processing Platform.

В разделе 2 указаны требования к программно-техническим средствам, необходимым для работы 3i Big Data Analytics Processing Platform.

В разделе 3 указывается описание задач, решаемых 3i Big Data Analytics Processing Platform, даются сведения об используемых технологиях.

В разделе 4 даются сведения об установке 3i Big Data Analytics Processing Platform.

В разделе 5 приводится описание интерфейса программирования 3i Big Data Analytics Processing Platform.

В разделе 6 даются сведения о входных и выходных данных 3i Big Data Analytics Processing Platform.

В разделе 7 приводятся основные сообщения оператору при работе с 3i Big Data Analytics Processing Platform.

По всем вопросам, связанным с использованием СПО «3i Big Data Analytics Processing Platform» можно обращаться по электронной почте support@dss-lab.ru или по телефону +7 (495) 645-44-70 по будним дням с 10 до 18 часов, время московское.

СОДЕРЖАНИЕ

1. НАЗНАЧЕНИЕ ПРОГРАММЫ

Специальное программное обеспечение «3i Big Data Analytics Processing Platform» предназначено для решения задач аналитической обработки структурированной информации пользователей социальных сетей. 3i Big Data Analytics Processing Platform используется в качестве отдельного модуля, встраиваемого в другое программное обеспечение, предоставляя разработчику соответствующий функционал API. API (сокр. англ. Application Programming Interface, интерфейс программирования приложений, интерфейс прикладного программирования) – набор готовых классов, процедур, функций, структур и констант, предоставляемых приложением для использования во внешних программных продуктах.

API 3i Big Data Analytics Processing Platform относится к классу REST (сокр. англ. Representational State Transfer, передача репрезентативного состояния) – метод взаимодействия компонентов распределённого приложения в сети Интернет, при котором вызов удаленной процедуры представляет собой обычный HTTP-запрос (обычно GET или POST; такой запрос называют REST-запрос), а необходимые данные передаются в качестве параметров запроса. Этот способ является альтернативой более сложным методам, таким как SOAP, CORBA и RPC.

К основным преимуществам приложения, предоставляющего HTTP REST API в качестве инструмента доступа к функциональным возможностям, можно отнести:

- надёжность (за счет отсутствия необходимости сохранять информацию о состоянии клиента, которая может быть утеряна);
- производительность (за счет использования кэша);
- прозрачность системы взаимодействия;
- легкость внесения изменений;
- масштабируемость.

2. УСЛОВИЯ ПРИМЕНЕНИЯ

2.1. Для функционирования 3i Big Data Analytics Processing Platform необходим кластер вычислительных систем, минимальные параметры каждой из которых:

– CPU частотой не менее 2 ГГц (8 физических вычислительных ядер на 1 процессор);

– ОЗУ 16 ГБ;

– LAN 1 Гбит/с;

– дисковая подсистема совокупной емкостью не менее 1 Тб.

Количество вычислительных систем в кластере зависит от требований заказчика.

2.2. Для функционирования 3i Big Data Analytics Processing Platform на вычислительной системе должно быть установлено следующее общее программное обеспечение:

– операционная система – Linux Ubuntu 14.04 LTS или Linux CentOS 7.15;

– пакет типового программного обеспечения операционной системы семейства Linux (post-install) – «Everything» в случае использования Linux CentOS 7.15 (http://isoredirect.centos.org/centos/7/isos/x86_64/CentOS-7-x86_64-Everything-1503-01.iso);

– реализация виртуальной машины, необходимая для исполнения Java-приложений – Oracle Java Runtime Environment (JRE) версии 8 или выше;

– набор инструментов для удаленного контроля и обмена данными с сетевыми компьютерами – OpenSSH Server 7.1 или выше;

– набор программ, реализующих протокол SMB в среде Unix – Samba 4.3.1 или выше.

3. ОПИСАНИЕ ЗАДАЧИ

3.1. Технологии аналитической обработки структурированной информации пользователей социальных сетей

Возможности 3i Big Data Analytics Processing Platform базируются на технологиях Apache Spark – независимой, быстрой платформе для распределённых вычислений, поддерживающей обработку данных по модели MapReduce, Pregel, GraphX.

Ключевым понятием системы Spark является распределённый набор данных (англ. resilient distributed datasets или RDD). RDD – это коллекция неизменяемых объектов, распределенных по множеству узлов, которые могут обрабатываться в параллельном режиме. Эти коллекции устойчивы, потому что в случае потери части набора данных они могут легко восстанавливаться. Процесс восстановления части набора данных опирается на механизм отказоустойчивости, поддерживающий родословную RDD – информацию, которая позволяет восстанавливать часть набора данных с помощью процесса, в результате которого эти данные были получены. Представление данных в виде RDD позволяет временно сохранять данные, необходимые для вычислений, в оперативной памяти узла кластера для уменьшения времени доступа. Возможность проводить кластерные вычисления в памяти позволяет более эффективно реализовывать итеративные алгоритмы и решать задачи интерактивного извлечения данных.

Быстрое восстановление потерянной части данных и сохранение информации данных в памяти являются основными преимуществами системы Spark перед другими системами распределённых вычислений. Spark реализован на языке Scala и использует его в качестве среды разработки приложений. Этот язык позволяет легко манипулировать распределенными наборами данных как локальными объектами. RDD представляет собой объект Scala, который может создаваться из файла или же путём преобразования из другой RDD. Приложения в системе осуществляют операции над данными – RDD, выполняемые на одном

узле или параллельно на наборе узлов. Каждое приложение, написанное под систему Spark, представляет собой цепочку операций, которые преобразовывают исходную RDD к некоторым конечным данным (не обязательно RDD). Операции над данными делятся на два вида: преобразования и действия. Преобразования создают новый RDD основываясь на уже существующем RDD. Действия либо возвращают некоторое значение в программу, запускающую это приложение, причём это значение является результатом некоторого вычисления проводимого над какой-либо RDD, либо сохраняют некоторые данные в памяти или на диске. Технологии Apache Spark обеспечивают:

– хранение информации в виде сбое-устойчивых распределённых наборов данных (RDD);

– совместимость с Hadoop (HDFS, Hbase, SequenceFiles и т.п.);

– кэширование RDD в оперативной памяти узлов кластера;

– высокую скорость обработки интерактивных запросов (до 100 раз быстрее Hadoop);

– возможность манипулировать данными с помощью различных параллельных операций (таких как map и reduce).

3i Big Data Analytics Processing Platform предоставляет пользователю развитой алгоритмический функционал аналитической обработки структурированной информации пользователей социальных сетей:

– Расчёт популярности. Популярность вершины/пользователя определяется количеством вершин/пользователей, на которые она может оказывать прямое воздействие.

– Расчёт хабовости. Хабовость вершины/пользователя определяется количеством вершин/пользователей, которые могут оказывать на нее непосредственное воздействие (иначе – количеством вершин/пользователей, на которые она ссылается).

– Расчет близости. Близость вершины/пользователя – показатель, характеризующий то, насколько данная вершина/пользователя находится близко по цепочкам связей ко всем остальным вершинам/пользователям социальной сети.

– Расчет посредничества. Посредничество вершины/пользователя – вероятность того, что выбранная вершина/пользователь будет вовлечена во все коммуникации, проводимые по кратчайшим путям между двумя другими вершинами/пользователями (может быть осуществлено управление коммуникациями между другими вершинами/пользователями).

– Расчет авторитетности (PageRank). PageRank – мера центральности вершин/пользователей по собственному вектору, широко используемая в сетевом анализе. Алгоритм применяется к коллекции вершин/пользователей, связанных гиперссылками, и назначает каждому из них некоторое численное значение, измеряющее его относительную важность среди остальных вершин/пользователей.

– Расчет активности. Активность вершины/пользователя можно определить, как количество совершенных вершиной/пользователем действий. Для каждого действия может быть определен свой вес.

– Расчёт прямого влияния. Прямое влияние определяется как Соотношение активности выбранной вершины/пользователя по отношению к соседним вершинам/пользователям.

3.2. Структура 3i Big Data Analytics Processing Platform

Обобщенная структурная схема 3i Big Data Analytics Processing Platform представлена на рисунке 3.1

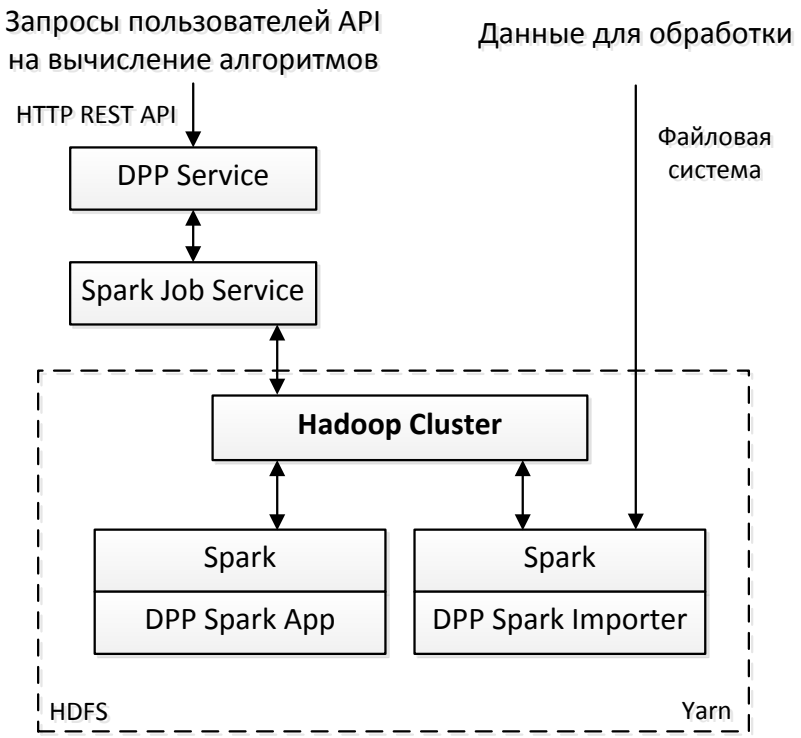

Рисунок 3.1 – Обобщенная структурная схема 3i Big Data Analytics Processing Platform

Структура 3i Big Data Analytics Processing Platform включает следующие компоненты:

– DPP Service – обрабатывает запросы пользователей API на вычисление алгоритмов, является сервисом, работающим внутри веб-сервера Apache Tomcat;

– Spark Job Server – предоставляет функциональность для использования Spark как сервиса (кеширование данных между отдельными вычислениями, отсутствие затрат на постоянное пересоздание контекста вычисления);

– Hadoop Cluster – совокупность вычислительных узлов, объединённых в кластер под управлением Hadoop Yarn;

– Spark – система для распределённой обработки данных;

– DPP Spark Importer – реализует функциональность парсинга, обработки, и добавления данных из внешних источников при помощи Apache Spark в рамках кластера Hadoop;

– DPP Spark App – модуль, реализующий распределённые вычисления алгоритмов при помощи Apache Spark в рамках кластера Hadoop.

Технологическая цепочка аналитической обработки данных представлена на рисунке 3.2.

Рисунок 3.2 – Технологическая цепочка аналитической обработки данных

4. ВЫЗОВ И ЗАГРУЗКА ПРОГРАММЫ

3i Big Data Analytics Processing Platform устанавливается на кластер из нескольких вычислительных машин, среди которых выделяется одна – master (организует работу остальных вычислительных машин кластера и может сам предоставлять вычислительные ресурсы наряду со slave), остальные вычислительные машины называются slave (предоставляют вычислительные ресурсы кластеру). Распределение модулей 3i Big Data Analytics Processing Platform в процессе установки по master и slave следующее:

- кластер Hadoop под управлением YARN (устанавливается на master и на slave);
- вычислительная среда Apache Spark (устанавливается на master);
- Spark Job Server – система управления задачами в Apache Spark (устанавливается на master);
- DPPSparkApp – модуль, реализующий распределённые вычисления алгоритмов на кластере Hadoop при помощи Apache Spark (устанавливается на master);
- DPPSparkImporterApp – модуль, реализующий импорт данных в систему из внешних источников (устанавливается на master);
- DPPService — модуль, предоставляющий внешнее API для выполнения расчётов алгоритмов (устанавливается на master).

Далее представлены этапы установки модулей 3i Big Data Analytics Processing Platform с указанием пользовательских команд в консоли операционной системы.

4.1. Подготовка к установке 3i Big Data Analytics Processing Platform

1) Создать пользователя:

```
sudo useradd -d /home/dpp -m dpp
```

```
sudo passwd dpp
```

2) Скопировать дистрибутив системы в домашнюю директорию пользователя dpp:

```
sudo cp /path/to/installer/Installer.tar.gz /home/dpp/
```

где /path/to/Installer – путь до директории с дистрибутивом системы.

3) Установить права на архив пользователю dpp:

```
sudo chown dpp:dpp /home/dpp/Installer.tar.gz
```

4) Зайти в систему как пользователь dpp:

```
su dpp
cd ~
```

5) Извлечь содержимое дистрибутива:

```
tar -xvf Installer.tar.gz
```

6) Распаковать архивы на жесткий диск:

6.1) Для master:

```
tar -xvf ~/Installer/spark-1.5.2-bin-hadoop2.6.tgz -C ~
tar -xvf ~/Installer/hadoop-2.7.1.tar.gz -C ~
tar -xvf ~/Installer/apache-tomcat-8.0.28.tar.gz -C ~
tar -xvf ~/Installer/Importer.tar.gz -C ~
mkdir job-server
tar -xvf ~/Installer/job-server.tar.gz -C ~/job-server
```

6.2) Для slave:

```
tar -xvf ~/Installer/hadoop-2.7.1.tar.gz -C ~
```

4.2. Установка и настройка 3i Big Data Analytics Processing Platform

4.2.1. Настройка SSH доступа от master к slave

1) На вычислительной машине master выполнить следующие команды:

```
ssh-keygen -t rsa
ssh-copy-id -i ~/.ssh/id_rsa.pub dpp@master
```

где *master* – либо ip адрес машины master, либо dns псевдоним, указанный в начале файла hosts машины master.

2) Далее для каждого slave выполнить на машине master команду:

```
ssh-copy-id -i ~/.ssh/id_rsa.pub dpp@slave
```

где *slave* – либо ip адрес машины slave, либо dns псевдоним, указанный в начале файла hosts машины master.

4.2.2. Настройка Hadoop (на master и slave)

1) В файле ~/.bashrc дописать настройки переменных среды:

```
export HADOOP_HOME=/home/dpp/hadoop-2.7.1
export HADOOP_CONF_DIR=/home/dpp/hadoop-2.7.1/etc/hadoop
export JAVA_HOME=/path/to/java8
```

где /path/to/java8 — путь до домашней директории Java 8.

2) Создание рабочих директорий Hadoop. Указать директорию для временных файлов:

```
mkdir ~/tmp/
mkdir ~/tmp/hadoop
```

3) Создать папки, в которых будут храниться данные hdfs:

```
mkdir ~/hdfs
```

4) Настройка hadoop-env.sh

В файле ~/hadoop-2.7.1/etc/hadoop/hadoop-env.sh явно указать переменную среды JAVA_HOME:

```
export JAVA_HOME="/path/to/java8"
```

где /path/to/java8 — путь до домашней директории Java 8.

5) Настройки core-site.xml

В файле ~/hadoop-2.7.1/etc/hadoop/core-site.xml нужно установить путь до папки, в которую будут сохраняться временные файлы, и адрес интерфейса hdfs машины master:

```
<property>
 <name>hadoop.tmp.dir</name>
 <value>/home/dpp/tmp/hadoop</value>
</property>
<property>
 <name>fs.defaultFS</name>
 <value>hdfs://master:9000</value>
</property>
```

где *master* — либо ip адрес машины master, либо dns псевдоним машины master, указанный в начале файла hosts текущей вычислительной машины.

6) Настройки mapred-site.xml

Создать файл cp hadoop-2.7.1/etc/hadoop/mapred-site.xml.template hadoop-2.7.1/etc/hadoop/mapred-site.xml и установить в нём адрес интерфейса job tracker машины master:

```
<property>
 <name>mapred.job.tracker</name>
 <value>master:54311</value>
</property>
```

где *master* — либо ip адрес машины master, либо dns псевдоним машины master, указанный в начале файла hosts текущей машины.

7) Настройка hdfs-site.xml

7.1) Для master:

В файле ~/hadoop-2.7.1/etc/hadoop/hdfs-site.xml установить уровень репликации данных и директории, в которых будут храниться данные hdfs:

```
<property>
 <name>dfs.replication</name>
 <value>1</value>
</property>
<property>
 <name>dfs.namenode.name.dir</name>
 <value>/storage/hadoopTmp/namenode</value>
```

```
</property>
<property>
 <name>dfs.datanode.data.dir</name>
 <value>/storage/hadoopTmp/datanode</value>
</property>
```

7.2) Для slave:

В файле ~/hadoop-2.7.1/etc/hadoop/hdfs-site.xml установить директории, в которых будут храниться данные hdfs:

```
<property>
 <name>dfs.namenode.name.dir</name>
 <value>/home/dpp/hdfs/namenode</value>
</property>
<property>
 <name>dfs.datanode.data.dir</name>
 <value>/home/dpp/hdfs/datanode</value>
</property>
```

8) Настройка capacity-scheduler.xml

В файле ~/hadoop-2.7.1/etc/hadoop/capacity-scheduler.xml задать следующие настройки:

```
yarn.scheduler.capacity.maximum-am-resource-percent – 100
yarn.scheduler.capacity.root.default.user-limit-factor – 10
```

9) Настройка yarn-site.xml

9.1) В файле ~/hadoop-2.7.1/etc/hadoop/yarn-site.xml установить:

```
<property>
 <name>yarn.nodemanager.sleep-delay-before-sigkill.ms</name>
 <value>30000</value>
</property>
<property>
 <name>yarn.nodemanager.process-kill-wait.ms</name>
 <value>30000</value>
</property>
<property>
 <name>yarn.scheduler.minimum-allocation-mb</name>
 <value>512</value>
 <description>
```

9.2) Минимальное количество оперативной памяти, выделяемое для одного контейнера Hadoop (в терминах Spark контейнер Hadoop – это либо драйвер, либо исполнитель):

```
  </description>
</property>
<property>
  <name>yarn.nodemanager.resource.memory-mb</name>
  <value>14336</value>
  <description>
```

9.3) Общее количество оперативной памяти доступной на машине для node Hadoop:

```
  </description>
</property>
<property>
  <name>yarn.nodemanager.vmem-check-enabled</name>
  <value>false</value>
</property>
<property>
  <name>yarn.nodemanager.vmem-pmem-ratio</name>
  <value>4</value>
</property>
<property>
  <name>yarn.nodemanager.resource.cpu-vcores</name>
  <value>8</value>
  <description>
```

9.4) Количество доступных виртуальных ядер (одно ядро выделяется на один контейнер):

```
  </description>
</property>
<property>
  <name>yarn.nodemanager.aux-services</name>
  <value>mapreduce_shuffle</value>
</property>
<property>
  <name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
  <value>org.apache.hadoop.mapred.ShuffleHandler</value>
</property>
<property>
  <name>yarn.resourcemanager.scheduler.address</name>
```

```
  <value>master:8030</value>
</property>
<property>
  <name>yarn.resourcemanager.address</name>
  <value>master:8032</value>
  <description>
```

9.5) Адрес resource manager машины master:

```
  </description>
</property>
<property>
  <name>yarn.resourcemanager.webapp.address</name>
  <value>master:8088</value>
  <description>
```

9.6) Адрес web UI resource manager машины master:

```
  </description>
</property>
<property>
  <name>yarn.resourcemanager.resource-tracker.address</name>
  <value>master:8031</value>
</property>
<property>
  <name>yarn.resourcemanager.admin.address</name>
  <value>master:8033</value>
</property>
```

10) Так же на master в файле ~/hadoop-2.7.1/etc/hadoop/slaves нужно задать список адресов slave, которые master будет самостоятельно запускать (если сам master предоставляет кластеру вычислительные ресурсы, его так же надо включить в этот список):

master

slave1

slave2

Где master, slave1, slave2 и т.д. – либо ip адреса соответствующих машин, либо dns псевдонимы этих машины, указанные в начале файла hosts машины master.

4.2.3. Настройки Samba (на master и slave)

1) Создаём папку на master и на slave, с которой будет работать Samba (данная папка на машине master будет открыта для общего доступа, в результате чего slave смогут с ней работать как с локальной):

```
mkdir ~/shareFolder
```

2) На master нужно установить и настроить сервер Samba.

2.1) Устанавливаем сервер Samba из репозитория:

```
sudo apt-get install samba
```

2.2) Открываем конфигурационный файл Samba с правами root:

```
sudo nano /etc/samba/smb.conf
```

В данном файле указать значение следующих параметров:

```
   pam password change = yes
   map to guest = bad user
   usershare allow guests = yes
[importer]
   comment = For Importer To HDFS
   path=/home/dpp/shareFolder
   guest ok = yes
   read only = no
   create mask = 0777
   directory mask = 0777
   force user = nobody
   force group = nogroup
```

2.3) Перезагружаем сервер samba:

```
sudo service smbd restart
```

3) На slave нужно установить утилиту cifs-utils:

```
sudo apt-get install cifs-utils
```

4.2.4. Настройки Spark (на master)

1) Создать файл настроек спарка:

```
cp spark-1.5.2-bin-hadoop2.6/conf/spark-env.sh.template spark-1.5.2-bin-hadoop2.6/conf/spark-env.sh
```

2) Указать в нём путь до директории с конфигурационными файлами Hadoop:

```
HADOOP_CONF_DIR=/home/dpp/hadoop-2.7.1/etc/hadoop/
```

4.2.5. Настройки Spark Job Server (на master)

1) Создать директорию для хранения логов:

```
mkdir ~/job-server/log
```

2) Создать директорию для хранения временных файлов:

```
mkdir ~/job-server/tmp
```

3) В файле ~/job-server/settings.sh указать значение следующих параметров:

3.1) Задать имя пользователя, от имени которого будет запускаться spark job server:

```
APP_USER=dpp
```

3.2) Задать директорию, в которой находятся файлы spark job server:

```
INSTALL_DIR=/home/dpp/job-server
```

3.3) Задать директорию, в которой spark job server будет сохранять файлы логов:

```
LOG_DIR=/home/dpp/job-server/log
```

3.4) Если нужно, указать количество памяти, выделяемое для spark job server:

```
JOBSERVER_MEMORY=1G
```

3.5) Задать путь до директории, в которую распакован Spark:

```
SPARK_HOME=/home/dpp/spark-1.5.2-bin-hadoop2.6
```

3.6) Задать путь до директории, в которой находятся конфигурационные файлы Spark:

```
SPARK_CONF_DIR=/home/dpp/spark-1.5.2-bin-hadoop2.6/conf
```

4) В файле ~/job-server/local.conf указать значение следующих параметров:

4.1) Директории для хранения временных файлов:

```
spark.jobserver.jar-store-rootdir = /home/dpp/job-server/tmp
spark.jobserver.filedao.rootdir = /home/dpp/job-server/tmp
spark.jobserver.datadao.rootdir = /home/dpp/job-server/tmp
spark.jobserver.sqldao.rootdir = /home/dpp/job-server/tmp
```

4.2) Количество экзекуторов (равно количеству ядер), используемых для вычислений алгоритмов:

```
spark.context-settings.executor.instances = 6
```

4.3) Количество памяти, используемое каждым исполнителем (суммарное количество памяти равно количеству исполнителей, умноженному на количество памяти у одного экзекутора):

```
spark.context-settings.memory-per-node = 3g
```

4.4) Директорию, в которую установлен Spark (кавычки обязательны):

```
spark.home = "/home/dpp/spark-1.5.2-bin-hadoop2.6"
```

4.2.6. Настройка DPPService и Apache Tomcat (на master)

1) Скопировать библиотеку в рабочую директорию Tomcat:

```
cp ~/Installer/DPPService.war ~/apache-tomcat-8.0.28/webapps/
```

4.3. Запуск 3i Big Data Analytics Processing Platform

1) Запуск slave:

Монтируем удалённую папку на локальную машину в указанную директорию

```
sudo mount -t cifs //master/importer ~/shareFolder
```

2) Запуск master:

2.1) Запуск кластера Hadoop.

Перед первым запуском кластера нужно форматировать hdfs:

```
~/hadoop-2.7.1/bin/hadoop namenode -format
```

Команды для запуска yarn и hdfs:

```
$HADOOP_HOME/sbin/start-dfs.sh
$HADOOP_HOME/sbin/start-yarn.sh
```

2.2) Запуск Spark Job Server.

```
~/job-server/server_start.sh
```

При первом запуске нужно загрузить библиотеку DPPSparkApp в систему (после запуска Spark Job Server):

```
curl -i --data-binary "@/home/dpp/Installer/JobServerApi-assembly-1.0.jar" master:8090/jars/DPPSparkAppJar
```

2.3) Запуск сервера Apache Tomcat.

```
~/apache-tomcat-8.0.28/bin/startup.sh
```

2.4) Запуск DPPSparkImporterApp.

2.5) Мониторинг и управление параметрами модулей:

1) UI Resource Manager кластера:

```
master:8088/cluster
```

2) UI Spark Job Server:

```
master:8090
```

3) API для вычисления алгоритмов:

```
master:8080/DPPService
```

где *master* — либо ip адрес вычислительной машины master, либо dns псевдоним машины master, указанный в начале файла hosts текущей вычислительной машины.

5. ВЫПОЛНЕНИЕ ПРОГРАММЫ

5.1. Описание функционала API

Функции API 3i Big Data Analytics Processing Platform условно разделены на два крупных специализированных блока:

– Блок аналитической обработки – обеспечивается возможность аналитической обработки поступивших данных совокупностью алгоритмического обеспечения системы;

– Блок обмена данными – обеспечивается возможность импорта структурированной информации и экспорта результатов ее аналитической обработки.

Далее для каждого блока приведено отдельное описание функций, типов полей, обработок и ошибок API.

5.1.1. Блок аналитической обработки

Обращение к блоку аналитической обработки происходит посредством HTTP-запросов. Тело запросов и ответов представлено в JSON формате.

Таблица 5.1.1.1 Полный список функций блока аналитической обработки

Наименование функции	**Пример запроса**
Алгоритмы	
Расчёт популярности	PUT /algorithms/popularity
Расчёт хабовости	PUT /algorithms/hub
Расчёт близости	PUT /algorithms/proximity
Расчёт посредничества	PUT /algorithms/mediation
Расчёт PageRank	PUT /algorithms/pageRank
Расчёт активности	PUT /algorithms/activity

Наименование функции	Пример запроса
Расчёт прямого влияния	PUT /algorithms/directEffect
Результаты обработки	
Получение списка выполненных расчётов	GET /job
Получение состояния расчёта	GET /job/{jobId}

5.1.1.1. Расчет популярности

Запрос

```
PUT /algorithms/popularity
```

Параметры запроса:

orderByValue – указание сортировки результатов по значениям алгоритма (true) или по id пользователей (false), значение по умолчанию – true;

ascending – указание сортировки результатов по возрастанию (true) или по убыванию (false), значение по умолчанию – false;

limit – количество возвращаемых элементов в порядке сортировки, значение по умолчанию – 50;

network – значение фильтра по сети (возможные на данный момент значения: ALL, vk.com , twitter.com), значение по умолчанию – ALL.

Пример запроса:

```
PUT
/algorithms/popularity?orderByValue=true&ascending=false&limit=50&network=VK.COM
```

Успешный ответ

Таблица 5.1.1.2 Параметры ответа

Параметр	Тип данных	Обязательность	Описание
status	string	Обязательный	Статус выполнения расчёта.
result.jobId	string	Обязательный	Id job-a для мониторинга состояния выполнения расчёта.
result.context	string	Обязательный	Имя контекста, в котором выполняется расчёт.

Пример успешного ответа:

```
{
  "status": "STARTED",
  "result": {
      "jobId": "<string>",
      "context": "<string>"
  }
}
```

Ответ с ошибкой

Таблица 5.1.1.3 Коды состояния HTTP

Код	Описание
404	«Not Found». Сервер понял запрос, но не нашёл соответствующего ресурса по указанному URL.

Пример ответа с ошибкой:

```
<!DOCTYPE
 html><html><head><title>Apache Tomcat/8.0.28 - Error report</title><style
type="text/css">H1 {font-family:Tahoma,Arial,sans-serif;color:white;background-
color:#525D76;font-size:22px;} H2 {font-family:Tahoma,Arial,sans-
serif;color:white;background-color:#525D76;font-size:16px;}
 H3 {font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;font-
size:14px;} BODY {font-family:Tahoma,Arial,sans-serif;color:black;background-color:white;} B
{font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;} P {font-
family:Tahoma,Arial,sans-serif;background:white;color:black;font-size:12px;}
A {color : black;}A.name {color : black;}.line {height: 1px; background-color: #525D76; border:
none;}</style> </head><body><h1>HTTP Status 404 - Not Found</h1><div
class="line"></div><p><b>type</b> Status report</p><p><b>message</b> <u>Not
Found</u></p><p><b>description</b><u>The requested resource is not
available.</u></p><hr class="line"><h3>Apache Tomcat/8.0.28</h3></body></html>
```

5.1.1.2. Расчет хабовости

Запрос

```
PUT   /algorithms/hub
```

Параметры запроса:

orderByValue – указание сортировки результатов по значениям алгоритма (true) или по id пользователей (false), значение по умолчанию – true;

ascending – указание сортировки результатов по возрастанию (true) или по убыванию (false), значение по умолчанию – false;

limit – количество возвращаемых элементов в порядке сортировки, значение по умолчанию – 50;

network – значение фильтра по сети (возможные на данный момент значения: ALL, vk.com , twitter.com), значение по умолчанию – ALL.

Пример запроса:

```
PUT /algorithms/hub?orderByValue=true&ascending=false&limit=50&network=vk.com
```

Успешный ответ

Таблица 5.1.1.4 Параметры ответа

Параметр	Тип данных	Обязательность	Описание
status	string	Обязательный	Статус выполнения расчёта.
result.jobId	string	Обязательный	Id job-a для мониторинга состояния выполнения расчёта.
result.context	string	Обязательный	Имя контекста, в котором выполняется расчёт.

Пример успешного ответа:

```
{
  "status": "STARTED",
  "result": {
      "jobId": "<string>",
      "context": "<string>"
  }
}
```

Ответ с ошибкой

Таблица 5.1.1.5 Коды состояния HTTP

Код	Описание
404	«Not Found». Сервер понял запрос, но не нашёл соответствующего ресурса по указанному URL.

Пример ответа с ошибкой:

```
<!DOCTYPE
 html><html><head><title>Apache Tomcat/8.0.28 - Error report</title><style
type="text/css">H1 {font-family:Tahoma,Arial,sans-serif;color:white;background-
color:#525D76;font-size:22px;} H2 {font-family:Tahoma,Arial,sans-
serif;color:white;background-color:#525D76;font-size:16px;}
 H3 {font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;font-
size:14px;} BODY {font-family:Tahoma,Arial,sans-serif;color:black;background-color:white;} B
{font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;} P {font-
family:Tahoma,Arial,sans-serif;background:white;color:black;font-size:12px;}
A {color : black;}A.name {color : black;}.line {height: 1px; background-color: #525D76; border:
none;}</style> </head><body><h1>HTTP Status 404 - Not Found</h1><div
class="line"></div><p><b>type</b> Status report</p><p><b>message</b> <u>Not
Found</u></p><p><b>description</b><u>The requested resource is not
available.</u></p><hr class="line"><h3>Apache Tomcat/8.0.28</h3></body></html>
```

5.1.1.3. Расчет близости

Запрос

```
PUT   /algorithms/proximity
```

Параметры запроса:

orderByValue – указание сортировки результатов по значениям алгоритма (true) или по id пользователей (false), значение по умолчанию – true;

ascending – указание сортировки результатов по возрастанию (true) или по убыванию (false), значение по умолчанию – false;

limit – количество возвращаемых элементов в порядке сортировки, значение по умолчанию – 50;

network – значение фильтра по сети (возможные на данный момент значения: ALL, vk.com , twitter.com), значение по умолчанию – ALL;

maxIterations – указание максимального количества итераций алгоритма поиска путей на графе (при увеличении возрастает точность, но производительность сильно падает), значение по умолчанию – 5.

Пример запроса:

```
PUT /algorithms/proximity?orderByValue=true&ascending=false&limit=50&network=vk.com
```

Успешный ответ

Таблица 5.1.1.6 Параметры ответа

Параметр	**Тип данных**	**Обязательность**	**Описание**
status	string	Обязательный	Статус выполнения расчёта.
result.jobId	string	Обязательный	Id job-a для мониторинга состояния выполнения расчёта.
result.context	string	Обязательный	Имя контекста, в котором выполняется расчёт.

Пример успешного ответа:

```
{
  "status": "STARTED",
  "result": {
      "jobId": "<string>",
      "context": "<string>"
  }
}
```

Ответ с ошибкой

Таблица 5.1.1.7 Коды состояния HTTP

Код	Описание
404	«Not Found». Сервер понял запрос, но не нашёл соответствующего ресурса по указанному URL.

Пример ответа с ошибкой:

```
<!DOCTYPE
 html><html><head><title>Apache Tomcat/8.0.28 - Error report</title><style
type="text/css">H1 {font-family:Tahoma,Arial,sans-serif;color:white;background-
color:#525D76;font-size:22px;} H2 {font-family:Tahoma,Arial,sans-
serif;color:white;background-color:#525D76;font-size:16px;}
 H3 {font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;font-
size:14px;} BODY {font-family:Tahoma,Arial,sans-serif;color:black;background-color:white;} B
{font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;} P {font-
family:Tahoma,Arial,sans-serif;background:white;color:black;font-size:12px;}
A {color : black;}A.name {color : black;}.line {height: 1px; background-color: #525D76; border:
none;}</style> </head><body><h1>HTTP Status 404 - Not Found</h1><div
class="line"></div><p><b>type</b> Status report</p><p><b>message</b> <u>Not
Found</u></p><p><b>description</b><u>The requested resource is not
available.</u></p><hr class="line"><h3>Apache Tomcat/8.0.28</h3></body></html>
```

5.1.1.4. Расчет посредничества

Запрос

```
PUT   /algorithms/mediation
```

Параметры запроса:

orderByValue – указание сортировки результатов по значениям алгоритма (true) или по id пользователей (false), значение по умолчанию – true;

ascending – указание сортировки результатов по возрастанию (true) или по

убыванию (false), значение по умолчанию – false;

limit – количество возвращаемых элементов в порядке сортировки, значение по умолчанию – 50;

network – значение фильтра по сети (возможные на данный момент значения: ALL, vk.com , twitter.com), значение по умолчанию – ALL;

maxIterations – указание максимального количества итераций алгоритма поиска путей на графе (при увеличении возрастает точность, но производительность сильно падает), значение по умолчанию – 5;

mediationPartitions – указание степени разбиения данных при вычислении посредничества (требуется увеличивать значение при росте количества данных для корректного размещения информационных блоков в оперативной памяти; при росте значения скорость вычисления уменьшается), значение по умолчанию – 800.

Пример запроса:

```
PUT
/algorithms/mediation?orderByValue=true&ascending=false&limit=50&network=vk.com&maxIterations=10&mediationPartitions=1000
```

Успешный ответ

Таблица 5.1.1.8 Параметры ответа

Параметр	**Тип данных**	**Обязательность**	**Описание**
status	string	Обязательный	Статус выполнения расчёта.
result.jobId	string	Обязательный	Id job-a для мониторинга состояния выполнения

Параметр	Тип данных	Обязательность	Описание
			расчёта.
result.context	string	Обязательный	Имя контекста, в котором выполняется расчёт.

Пример успешного ответа:

```
{
  "status": "STARTED",
  "result": {
      "jobId": "<string>",
      "context": "<string>"
  }
}
```

Ответ с ошибкой

Таблица 5.1.1.9 Коды состояния HTTP

Код	Описание
404	«Not Found». Сервер понял запрос, но не нашёл соответствующего ресурса по указанному URL.

Пример ответа с ошибкой:

```
<!DOCTYPE
 html><html><head><title>Apache Tomcat/8.0.28 - Error report</title><style
type="text/css">H1 {font-family:Tahoma,Arial,sans-serif;color:white;background-
color:#525D76;font-size:22px;} H2 {font-family:Tahoma,Arial,sans-
serif;color:white;background-color:#525D76;font-size:16px;}
 H3 {font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;font-
size:14px;} BODY {font-family:Tahoma,Arial,sans-serif;color:black;background-color:white;} B
```

```
{font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;} P {font-
family:Tahoma,Arial,sans-serif;background:white;color:black;font-size:12px;}
A {color : black;}A.name {color : black;}.line {height: 1px; background-color: #525D76; border:
none;}</style> </head><body><h1>HTTP Status 404 - Not Found</h1><div
class="line"></div><p><b>type</b> Status report</p><p><b>message</b> <u>Not
Found</u></p><p><b>description</b><u>The requested resource is not
available.</u></p><hr class="line"><h3>Apache Tomcat/8.0.28</h3></body></html>
```

5.1.1.5. Расчет PageRank

Запрос

```
PUT   /algorithms/pageRank
```

Параметры запроса:

orderByValue – указание сортировки результатов по значениям алгоритма (true) или по id пользователей (false), значение по умолчанию – true;

ascending – указание сортировки результатов по возрастанию (true) или по убыванию (false), значение по умолчанию – false;

limit – количество возвращаемых элементов в порядке сортировки, значение по умолчанию – 50;

network – значение фильтра по сети (возможные на данный момент значения: ALL, vk.com , twitter.com), значение по умолчанию – ALL;

maxIterations – указание максимального количества итераций алгоритма поиска путей на графе (при увеличении возрастает точность, но производительность сильно падает), значение по умолчанию – 5;

resetProb – указание значения коэффициента затухания, значение по умолчанию – 0.25;

tolerance – указание точности, значение по умолчанию – 0.01.

Пример запроса:

```
PUT
/algorithms/pageRank?orderByValue=true&ascending=false&limit=50&network=vk.com&maxIterations=10&resetProb=0.15&tolerance=0.001
```

Успешный ответ

Таблица 5.1.1.10 Параметры ответа

Параметр	**Тип данных**	**Обязательность**	**Описание**
status	string	Обязательный	Статус выполнения расчёта.
result.jobId	string	Обязательный	Id job-а для мониторинга состояния выполнения расчёта.
result.context	string	Обязательный	Имя контекста, в котором выполняется расчёт.

Пример успешного ответа:

```
{
  "status": "STARTED",
  "result": {
      "jobId": "<string>",
      "context": "<string>"
  }
}
```

Ответ с ошибкой

Таблица 5.1.1.11 Коды состояния HTTP

Код	Описание
404	«Not Found». Сервер понял запрос, но не нашёл соответствующего ресурса по указанному URL.

Пример ответа с ошибкой:

```
<!DOCTYPE
 html><html><head><title>Apache Tomcat/8.0.28 - Error report</title><style
type="text/css">H1 {font-family:Tahoma,Arial,sans-serif;color:white;background-
color:#525D76;font-size:22px;} H2 {font-family:Tahoma,Arial,sans-
serif;color:white;background-color:#525D76;font-size:16px;}
 H3 {font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;font-
size:14px;} BODY {font-family:Tahoma,Arial,sans-serif;color:black;background-color:white;} B
{font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;} P {font-
family:Tahoma,Arial,sans-serif;background:white;color:black;font-size:12px;}
A {color : black;}A.name {color : black;}.line {height: 1px; background-color: #525D76; border:
none;}</style> </head><body><h1>HTTP Status 404 - Not Found</h1><div
class="line"></div><p><b>type</b> Status report</p><p><b>message</b> <u>Not
Found</u></p><p><b>description</b><u>The requested resource is not
available.</u></p><hr class="line"><h3>Apache Tomcat/8.0.28</h3></body></html>
```

5.1.1.6. Расчет активности

Запрос

```
PUT   /algorithms/activity
```

Параметры запроса:

orderByValue – указание сортировки результатов по значениям алгоритма (true) или по id пользователей (false), значение по умолчанию – true;

ascending – указание сортировки результатов по возрастанию (true) или по убыванию (false), значение по умолчанию – false;

limit – количество возвращаемых элементов в порядке сортировки, значение по умолчанию – 50;

network – значение фильтра по сети (возможные на данный момент значения: ALL, vk.com , twitter.com), значение по умолчанию – ALL.

Пример запроса:

```
PUT /algorithms/activity?orderByValue=true&ascending=false&limit=50&network=vk.com
```

Успешный ответ

Таблица 5.1.1.12 Параметры ответа

Параметр	**Тип данных**	**Обязательность**	**Описание**
status	string	Обязательный	Статус выполнения расчёта.
result.jobId	string	Обязательный	Id job-a для мониторинга состояния выполнения расчёта.
result.context	string	Обязательный	Имя контекста, в котором выполняется расчёт.

Пример успешного ответа:

```
{
  "status": "STARTED",
  "result": {
      "jobId": "<string>",
      "context": "<string>"
  }
}
```

Ответ с ошибкой

Таблица 5.1.1.13 Коды состояния HTTP

Код	Описание
404	«Not Found». Сервер понял запрос, но не нашёл соответствующего ресурса по указанному URL.

Пример ответа с ошибкой:

```
<!DOCTYPE
 html><html><head><title>Apache Tomcat/8.0.28 - Error report</title><style
type="text/css">H1 {font-family:Tahoma,Arial,sans-serif;color:white;background-
color:#525D76;font-size:22px;} H2 {font-family:Tahoma,Arial,sans-
serif;color:white;background-color:#525D76;font-size:16px;}
 H3 {font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;font-
size:14px;} BODY {font-family:Tahoma,Arial,sans-serif;color:black;background-color:white;} B
{font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;} P {font-
family:Tahoma,Arial,sans-serif;background:white;color:black;font-size:12px;}
A {color : black;}A.name {color : black;}.line {height: 1px; background-color: #525D76; border:
none;}</style> </head><body><h1>HTTP Status 404 - Not Found</h1><div
class="line"></div><p><b>type</b> Status report</p><p><b>message</b> <u>Not
Found</u></p><p><b>description</b><u>The requested resource is not
available.</u></p><hr class="line"><h3>Apache Tomcat/8.0.28</h3></body></html>
```

5.1.1.7. Расчет прямого влияния

Запрос

```
PUT   /algorithms/directEffect
```

Параметры запроса:

orderByValue – указание сортировки результатов по значениям алгоритма (true) или по id пользователей (false), значение по умолчанию – true;

ascending – указание сортировки результатов по возрастанию (true) или по убыванию (false), значение по умолчанию – false;

limit – количество возвращаемых элементов в порядке сортировки, значение по

умолчанию – 50;

network – значение фильтра по сети (возможные на данный момент значения: ALL, vk.com , twitter.com), значение по умолчанию – ALL.

Пример запроса:

```
PUT
/algorithms/directEffect?orderByValue=true&ascending=false&limit=50&network=vk.com
```

Успешный ответ

Таблица 5.1.1.14 Параметры ответа

Параметр	**Тип данных**	**Обязательность**	**Описание**
status	string	Обязательный	Статус выполнения расчёта.
result.jobId	string	Обязательный	Id job-a для мониторинга состояния выполнения расчёта.
result.context	string	Обязательный	Имя контекста, в котором выполняется расчёт.

Пример успешного ответа:

```
{
  "status": "STARTED",
  "result": {
      "jobId": "<string>",
      "context": "<string>"
  }
}
```

Ответ с ошибкой

Таблица 5.1.1.15 Коды состояния HTTP

Код	Описание
404	«Not Found». Сервер понял запрос, но не нашёл соответствующего ресурса по указанному URL.

Пример ответа с ошибкой:

```
<!DOCTYPE
 html><html><head><title>Apache Tomcat/8.0.28 - Error report</title><style
type="text/css">H1 {font-family:Tahoma,Arial,sans-serif;color:white;background-
color:#525D76;font-size:22px;} H2 {font-family:Tahoma,Arial,sans-
serif;color:white;background-color:#525D76;font-size:16px;}
 H3 {font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;font-
size:14px;} BODY {font-family:Tahoma,Arial,sans-serif;color:black;background-color:white;} B
{font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;} P {font-
family:Tahoma,Arial,sans-serif;background:white;color:black;font-size:12px;}
A {color : black;}A.name {color : black;}.line {height: 1px; background-color: #525D76; border:
none;}</style> </head><body><h1>HTTP Status 404 - Not Found</h1><div
class="line"></div><p><b>type</b> Status report</p><p><b>message</b> <u>Not
Found</u></p><p><b>description</b><u>The requested resource is not
available.</u></p><hr class="line"><h3>Apache Tomcat/8.0.28</h3></body></html>
```

5.1.1.8. Получение списка выполненных расчетов

Запрос

```
GET   /job
```

Параметры запроса: без параметров.

Пример запроса:

```
GET   /job
```

Успешный ответ

Таблица 5.1.1.16 Параметры ответа

Параметр	**Тип данных**	**Обязательность**	**Описание**
status	FINISHED|RUNNING|ERROR	Обязательный	Статус выполнения расчёта.
jobId	string	Обязательный	Id job-a для мониторинга состояния выполнения расчёта.
duration	string	Обязательный	Длительность вычислений ("<double> sec").
startTime	date	Обязательный	Время начала вычислений.
result.message	string	Необязательный	Тип ошибки при вычислениях.

Пример успешного ответа:

```
{
  "status": "<FINISHED|RUNNING|ERROR>",
  "jobId": "<string>",
  "duration": "<string>",
  "startTime": "<date>",
  "result": { //только в случае status = ERROR
      "message": "<string>"
  }
}
```

Ответ с ошибкой

Таблица 5.1.1.17 Коды состояния HTTP

Код	Описание
404	«Not Found». Сервер понял запрос, но не нашёл соответствующего ресурса по указанному URL.

Пример ответа с ошибкой:

```
<!DOCTYPE
 html><html><head><title>Apache Tomcat/8.0.28 - Error report</title><style
type="text/css">H1 {font-family:Tahoma,Arial,sans-serif;color:white;background-
color:#525D76;font-size:22px;} H2 {font-family:Tahoma,Arial,sans-
serif;color:white;background-color:#525D76;font-size:16px;}
 H3 {font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;font-
size:14px;} BODY {font-family:Tahoma,Arial,sans-serif;color:black;background-color:white;} B
{font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;} P {font-
family:Tahoma,Arial,sans-serif;background:white;color:black;font-size:12px;}
A {color : black;}A.name {color : black;}.line {height: 1px; background-color: #525D76; border:
none;}</style> </head><body><h1>HTTP Status 404 - Not Found</h1><div
class="line"></div><p><b>type</b> Status report</p><p><b>message</b> <u>Not
Found</u></p><p><b>description</b><u>The requested resource is not
available.</u></p><hr class="line"><h3>Apache Tomcat/8.0.28</h3></body></html>
```

5.1.1.9. Получение состояния расчета

Запрос

```
GET   /job/{jobId}
```

Параметры запроса:

jobId – id расчёта, полученный из метода запуска расчёта.

Пример запроса:

```
GET /job/26d9bdfd-b29a-48b3-8629-5a6b47e412a6
```

Успешный ответ

Таблица 5.1.1.18 Параметры ответа

Параметр	**Тип данных**	**Обязательность**	**Описание**
status	FINISHED| RUNNING| ERROR	Обязательный	Статус выполнения расчёта.
jobId	string	Обязательный	Id job-а для мониторинга состояния выполнения расчёта.
duration	string	Обязательный	Длительность вычислений ("<double> sec").
startTime	date	Обязательный	Время начала вычислений.
result.data.{network} {userId}user.name	string	Необязательный	Имя пользователя.
result.data.{network} {userId}user.url	string	Необязательный	Ссылка на страницу пользователя.
result.data.{network} {userId}user.value	double	Необязательный	Значение показателя для данного пользователя.
result.centrality	double	Необязательный	Значение центральности

Параметр	Тип данных	Обязательность	Описание
			для всех пользователей.
result.message	string	Необязательный	Тип ошибки при вычислениях.

Пример успешного ответа:

```
{ //если job с таким id найден
  "status": "<FINISHED|RUNNING|ERROR>",
  "jobId": "<string>",
  "duration": "<string>",
  "startTime": "<date>",
  "result": {
     "data": { //только в случае status = FINISHED
        "{network}{userId}user": {
           "name": "<string>",
           "url": "<string>",
           "value": <double>
        }
     },
     "centrality": <double>, //только в случае status = FINISHED
     "message": "<string>" //только в случае status = ERROR
  }
}
```

Ответ с ошибкой

Таблица 5.1.1.19 Коды состояния HTTP

Код	Описание
404	«Not Found». Сервер понял запрос, но не нашёл соответствующего ресурса по указанному URL.

Пример ответа с ошибкой:

```
<!DOCTYPE
 html><html><head><title>Apache Tomcat/8.0.28 - Error report</title><style
type="text/css">H1 {font-family:Tahoma,Arial,sans-serif;color:white;background-
color:#525D76;font-size:22px;} H2 {font-family:Tahoma,Arial,sans-
serif;color:white;background-color:#525D76;font-size:16px;}
 H3 {font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;font-
size:14px;} BODY {font-family:Tahoma,Arial,sans-serif;color:black;background-color:white;} B
{font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;} P {font-
family:Tahoma,Arial,sans-serif;background:white;color:black;font-size:12px;}
A {color : black;}A.name {color : black;}.line {height: 1px; background-color: #525D76; border:
none;}</style> </head><body><h1>HTTP Status 404 - Not Found</h1><div
class="line"></div><p><b>type</b> Status report</p><p><b>message</b> <u>Not
Found</u></p><p><b>description</b><u>The requested resource is not
available.</u></p><hr class="line"><h3>Apache Tomcat/8.0.28</h3></body></html>
```

5.1.2. Блок обмена данными

Обращение к блоку обмена данными происходит посредством HTTP-запросов.

Таблица 5.1.2.1 Полный список функций блока обмена данными

Наименование функции	**Пример запроса**
Импортинг данных	./start-importer.sh
Остановка импортинга	touch

5.1.2.1. Импортинг данных

Запрос

```
./start-importer.sh
```

Параметры запроса:

host – указание IP-адреса sftp-сервера;

username – указание имени пользователя sftp-сервера;

password – указание пароля пользователя sftp-сервера;

sftp_dir – указание пути к папке, с которой производится импортинг данных на sftp-сервере (у значения обязательно необходимо поставить разделитель файловой системы);

need_connect – указание необходимо ли брать данные с sftp-сервера; true — да, false — нет.

del_remote – указание необходимости удаления файлов после прочтения на sftp-сервере: true – да, false – нет;

local_dir – указание пути к папке, которая монтирована на каждом узле кластера по определённому пути (у значения обязательно необходимо поставить разделитель файловой системы);

del_local – указание необходимости удаления файлов после прочтения с локальной папки: true – да, false – нет;

hdfs_path – указание пути к HDFS-папке, в которой лежит граф данных пользователей социальной сети (у значения обязательно необходимо поставить разделитель файловой системы);

unique_check – указание необходимости проверки данных на уникальность: true – да, false – нет;

interval – указание интервала времени в секундах, через который процесс импортинга повторится;

executor-memory – указание количества оперативной памяти узла кластера, которая выделяется на каждый исполнительный блок Spark;

num-executors – указание количества исполнительных блоков Spark;

spark.driver.memory – указание количества оперативной памяти узла кластера, которая выделяется для драйвера Spark.

Пример запроса:

```
./start_importer.sh host=10.250.83.242 username=searchplatform password=dsslab sftp_dir=/home/searchplatform/file_results_bk/ need_connect=true del_remote=false local_dir=/storage/tmp/local/ del_local=true hdfs_path=/data/graph_final_test/ unique_check=true interval=60 executor-memory=900m num-executors=3 spark.driver.memory=2g
```

Успешный ответ

В терминале операционной системы выводится информация о ходе выполнения процесса импортинга (операции, этапы выполнения).

```
15/11/16 13:32:00 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
15/11/16 13:32:01 INFO client.RMProxy: Connecting to ResourceManager at master/10.250.83.175:8032
15/11/16 13:32:01 INFO yarn.Client: Requesting a new application from cluster with 2 NodeManagers
15/11/16 13:32:01 INFO yarn.Client: Verifying our application has not requested more than the maximum memory capability of the cluster (8192 MB per container)
15/11/16 13:32:01 INFO yarn.Client: Will allocate AM container, with 2432 MB memory including 384 MB overhead
15/11/16 13:32:01 INFO yarn.Client: Setting up container launch context for our AM
15/11/16 13:32:01 INFO yarn.Client: Setting up the launch environment for our AM container
15/11/16 13:32:01 INFO yarn.Client: Preparing resources for our AM container
15/11/16 13:32:01 INFO yarn.Client: Uploading resource file:/home/dpp/spark-1.5.2-bin-hadoop2.6/lib/spark-assembly-1.5.2-hadoop2.6.0.jar -> hdfs://master:9000/user/dpp/.sparkStaging/application_1447668877899_0004/spark-assembly-1.5.2-hadoop2.6.0.jar
15/11/16 13:32:01 INFO yarn.Client: Uploading resource file:/home/dpp/Importer/DataManager-1.0-jar-with-dependencies.jar -> hdfs://master:9000/user/dpp/.sparkStaging/application_1447668877899_0004/DataManager-1.0-jar-with-dependencies.jar
15/11/16 13:32:02 INFO yarn.Client: Uploading resource file:/tmp/spark-a875367a-245c-45ef-aa41-7c1c6aaf4103/__spark_conf__3285666522929551474.zip -> hdfs://master:9000/user/dpp/.sparkStaging/application_1447668877899_0004/__spark_conf__3285666522929551474.zip
15/11/16 13:32:02 INFO spark.SecurityManager: Changing view acls to: dpp
15/11/16 13:32:02 INFO spark.SecurityManager: Changing modify acls to: dpp
15/11/16 13:32:02 INFO spark.SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users with view permissions: Set(dpp); users with modify permissions: Set(dpp)
15/11/16 13:32:02 INFO yarn.Client: Submitting application 4 to ResourceManager
15/11/16 13:32:02 INFO impl.YarnClientImpl: Submitted application application_1447668877899_0004
15/11/16 13:32:03 INFO yarn.Client: Application report for application_1447668877899_0004 (state: ACCEPTED)
15/11/16 13:32:03 INFO yarn.Client:
```

```
   client token: N/A
   diagnostics: N/A
   ApplicationMaster host: N/A
   ApplicationMaster RPC port: -1
   queue: default
   start time: 1447669922206
   final status: UNDEFINED
   tracking URL: http://master:8088/proxy/application_1447668877899_0004/
   user: dpp
15/11/16 13:32:04 INFO yarn.Client: Application report for application_1447668877899_0004
(state: ACCEPTED)
15/11/16 13:32:05 INFO yarn.Client: Application report for application_1447668877899_0004
(state: ACCEPTED)
15/11/16 13:32:06 INFO yarn.Client: Application report for application_1447668877899_0004
(state: RUNNING)
15/11/16 13:32:06 INFO yarn.Client:
   client token: N/A
   diagnostics: N/A
   ApplicationMaster host: 10.250.83.175
   ApplicationMaster RPC port: 0
   queue: default
   start time: 1447669922206
   final status: UNDEFINED
   tracking URL: http://master:8088/proxy/application_1447668877899_0004/
   user: dpp
15/11/16 13:32:07 INFO yarn.Client: Application report for application_1447668877899_0004
(state: RUNNING)
15/11/16 13:32:08 INFO yarn.Client: Application report for application_1447668877899_0004
(state: RUNNING)
```

Ответ с ошибкой

В терминале операционной системы будет отображено сообщение об ошибке процесса импортинга.

```
15/11/16 13:35:24 INFO yarn.Client: Application report for application_1447668877899_0004
(state: FINISHED)
15/11/16 13:35:24 INFO yarn.Client:
   client token: N/A
   diagnostics: N/A
   ApplicationMaster host: 10.250.83.175
   ApplicationMaster RPC port: 0
   queue: default
   start time: 1447669922206
   final status: SUCCEEDED
   tracking URL: http://master:8088/proxy/application_1447668877899_0004/
```

```
  user: dpp
15/11/16 13:35:24 INFO util.ShutdownHookManager: Shutdown hook called
15/11/16 13:35:24 INFO util.ShutdownHookManager: Deleting directory /tmp/spark-a875367a-245c-45ef-aa41-7c1c6aaf4103
```

5.1.2.2. Остановка импортинга

Запрос

```
touch
```

Параметры запроса:

/path/to/share/dir/stop – абсолютный путь к файлу с названием «stop», который создастся в монтированной директории.

Пример запроса:

```
touch /storage/tmp/local/stop
```

Успешный ответ

Файл с названием «stop» создастся в монтированной директории, что означает остановку процесса импортинга.

Ответ с ошибкой

В терминале операционной системы будет отображено сообщение об ошибке остановки процесса импортинга.

```
touch: cannot touch ‘/storage/tmp/local/stop’: Permission denied
```

6. ВХОДНЫЕ И ВЫХОДНЫЕ ДАННЫЕ

В качестве основного формата представления входных и выходных данных используется JSON (англ. JavaScript Object Notation) – текстовый формат обмена данными, основанный на JavaScript (происходит от подмножества языка стандарта ECMA-262 1999 года). Формат считается независимым от языка и может использоваться практически с любым языком программирования.

JSON-текст представляет собой (в закодированном виде) одну из двух структур:

– Набор пар ключ: значение. В различных языках это реализовано как объект, запись, структура, словарь, хэш-таблица, список с ключом или ассоциативный массив. Ключом может быть только строка, значением — любая форма.

– Упорядоченный набор значений. Во многих языках это реализовано как массив, вектор, список или последовательность.

В качестве значений в JSON используются структуры:

– Объект – это неупорядоченное множество пар ключ: значение, заключённое в фигурные скобки «{ }». Ключ описывается строкой, между ним и значением стоит символ «:». Пары ключ-значение отделяются друг от друга запятыми.

– Массив (одномерный) – это упорядоченное множество значений. Массив заключается в квадратные скобки «[ ]». Значения разделяются запятыми.

– Значение может быть строкой в двойных кавычках, числом, объектом, массивом, одним из литералов: true, false или null. Таким образом, структуры могут быть вложены друг в друга.

– Строка – это упорядоченное множество из нуля или более символов юникода, заключенное в двойные кавычки. Символы могут быть

указаны с использованием escape-последовательностей, начинающихся с обратной косой черты «\».

6.1. Входные данные

Входными данными являются структурированные наборы данных, представленных файлами в JSON формате. Размещение требуемых файлов осуществляется в общедоступной для вычислительных систем кластера папке (каждый узел имеет права на чтение и запись файлов). Данная папка должна быть монтирована в локальную файловую систему каждого вычислительного узла кластера. В сетевой папке содержатся файлы с определёнными названиями, и каждый файл состоит из набора JSON-подобных строк. Формат строки:

```
{“<string>”: “<string>”, “<string>”:“<string>”, …}
```

Примечание: каждая строка соответствует одному объекту. Этот объект должен записываться в одну строку. В конце строки после символа «}» сразу следует символ переноса строки.

Таблица 6.1.1 Название файлов с входными данными

Название файла	**Описание данных файла**
relations*	Связи между пользователями и «like»
tw_users*	Пользователи сети twitter.com
tw_posts*	Посты сети twitter.com
vk_users*	Пользователи сети vk.com
vk_posts*	Посты сети vk.com
vk_comments*	Комментарии сети vk.com

Таблица 6.1.2 Описание строк для файла “relations*”

Название свойства	**Формат**	**Описание**
creationDate	<yyyy-MM-ddTHH:mm:ssZ>	Дата создания связи
id	Строка	Идентификатор связи
id_1	<сеть{число}тип сущности>	Идентификатор источника действия
id_2	<сеть{число}тип сущности>	Идентификатор приёмника действия
modificationDate	<yyyy-MM-ddTHH:mm:ssZ>	Не используется в системе
type	<тип связи>	Тип связи

Пример строки файла:

{"creationDate":"2015-10-19T11:07:32Z","id":"4c4913c3edb296fe182a6558fb98e0bf8c077a87","id_1":"vk.com{256679198}user","id_2":"vk.com{217804805}user","modificationDate":"2015-10-14T16:20:34Z","type":"vk_user_follower"}

Таблица 6.1.3 Описание строк для файла “tw_users*”

Название свойства	**Формат**	**Описание**
creationDate	<yyyy-MM-ddTHH:mm:ssZ>	Дата создания пользователя
description	Строка	Не используется в системе
id	<сеть{число}тип сущности>	Идентификатор

Название свойства	Формат	Описание
		пользователя
location	Строка	Не используется в системе
modificationDate	<yyyy-MM-ddTHH:mm:ssZ>	Не используется в системе
name	Строка	Название пользователя
profile_link	Строка	Не используется в системе
screen_name	Строка	Не используется в системе
site	<сеть>	Название сети
url	Строка	Адрес страницы пользователя в социальной сети

Пример строки файла:

{"creationDate":"2015-05-29T08:28:06Z","description":"","id":"twitter.com{3229716043}user","location":"","modificationDate":"2015-08-13T22:47:16Z","name":"Жуматаева Айжан","profile_link":"","screen_name":"zhumataeva002","site":"twitter.com","url":"http:\/\/twitter.com\/zhumataeva002"}

Таблица 6.1.4 Описание строк для файла “tw_posts”

Название свойства	Формат	Описание
author	<сеть{число}тип сущности>	Идентификатор автора

Название свойства	**Формат**	**Описание**
creationDate	<yyyy-MM-ddTHH:mm:ssZ>	Дата создания поста
id	<сеть{число}тип сущности{число}тип сущности>	Идентификатор поста
language	Строка	Не используется в системе
modificationDate	<yyyy-MM-ddTHH:mm:ssZ>	Не используется в системе
parent	<сеть{число}тип сущности> или пустая строка	Идентификатор родительского поста
place	Строка	Не используется в системе
site	<сеть>	Название сети
source	Строка	Не используется в системе
text	Строка	Не используется в системе

Пример строки файла:

{"author":"twitter.com{3423623812}user","creationDate":"2015-09-11T12:39:55Z","id":"twitter.com{3423623812}user{642361938635304961}post","language":"ru","modificationDate":"2015-10-15T01:52:48Z","parent":"twitter.com{428172980}user{642361630878244864}post","place":"\n","site":"twitter.com","source":"Twitter Web Client","text":"@abbat15 Попал наконец...то"}

Таблица 6.1.5 Описание строк для файла “vk_users” (описание только используемых свойств)

Название свойства	**Формат**	**Описание**
creationDate	<yyyy-MM-ddTHH:mm:ssZ>	Дата создания пользователя
first_name	Строка	Имя пользователя
id	<сеть{число}тип сущности>	Идентификатор пользователя
last_name	Строка	Фамилия пользователя
site	<сеть>	Название сети
url	Строка	Ардес страницы пользователя в социальной сети

Пример строки файла:

{"about":"","bdate":"16.12.1991","city_id":1,"city_name":"Москва","country_id":1,"country_name":"Россия","creationDate":"2015-08-13T17:10:50Z","domain":"patriot_kz_01","fac_id":0,"fac_name":"","first_name":"Патриот","grad_year":0,"home_phone":"","id":"vk.com{276746392}user","interests":"","langs":"","last_name":"Казахстана","life_main_id":0,"mobile_phone":"","modificationDate":"2015-10-09T11:32:49Z","people_main_id":0,"political_id":0,"relation_id":0,"religion":"","sex_id":2,"site":"vk.com","status":"","uni_id":0,"uni_name":"","url":"http:\/\/vk.com\/patriot_kz_01"}

Таблица 6.1.6 Описание строк для файла “vk_posts”

Название свойства	**Формат**	**Описание**
author	<сеть{число}тип сущности>	Идентификатор автора

Название свойства	Формат	Описание
creationDate	<yyyy-MM-ddTHH:mm:ssZ>	Дата создания поста
id	<сеть{число}тип сущности{число}тип сущности>	Идентификатор поста
modificationDate	<yyyy-MM-ddTHH:mm:ssZ>	Не используется в системе
parent	<сеть{число}тип сущности> или пустая строка	Идентификатор родительского поста
site	<сеть>	Название сети
text	Строка	Не используется в системе

Пример строки файла:

{"author":"vk.com{300075576}user","creationDate":"2015-04-21T04:06:01Z","id":"vk.com{300075576}user{486}post","modificationDate":"2015-08-13T13:28:38Z","parent":"","site":"vk.com","text":"Под руководством Н. Назарбаева страна вышла на устойчивую траекторию развития"}

Таблица 6.1.7 Описание строк для файла “vk_comments”

Название свойства	Формат	Описание
author	<сеть{число}тип сущности>	Идентификатор автора
creationDate	<yyyy-MM-ddTHH:mm:ssZ>	Дата создания комментария
id	<сеть{число}тип сущности{число}тип	Идентификатор комментария

Название свойства	**Формат**	**Описание**
	сущности{число}тип сущности>	
modificationDate	<yyyy-MM-ddTHH:mm:ssZ>	Не используется в системе
parent	<сеть{число}тип сущности> или пустая строка	Идентификатор родительского поста
site	<сеть>	Название сети
text	Строка	Не используется в системе

Пример строки файла:

{"author":"vk.com{36014136}user","creationDate":"2015-02-15T10:29:55Z","id":"vk.com{26619469}group{132842}post{132970}comment","modificationDate":"2015-10-06T18:10:36Z","owner":"vk.com{26619469}group","parent":"vk.com{26619469}group{132842}post","site":"vk.com","text":"подскажите пожалуйста какие условия и требования выполнения норматива мастера спорта международного класса по вольной борьбе."}

Таблица 6.1.8 Дополнительно встречающиеся в файлах параметры строк

Название параметра	**Возможные значения**
<сеть>	twitter.com, vk.com
<тип связи>	vk_comment_liker, vk_post_liker, tw_post_link, vk_group_member, vk_user_follower, vk_group_link, vk_post_sharer, tw_user_link
<тип сущности>	user, group, post, comment

6.2. Выходные данные

Выходными данными являются структурированные наборы данных, представленные файлами в JSON формате. Данные файлы размещаются в распределённой файловой системе HDFS. Каждый файл соответствует сущности и состоит и набора JSON-подобных строк.

Таблица 6.2.1 Перечень файлов-сущностей

Название файла	**Описание данных файла**
actions	Действия, созданные пользователями (посты, репосты, комментарии, лайки)
create_links	Отношения создания действий
follow_links	Отношение следования пользователей
source_links	Отношение последствия для действий
users	Пользователи социальных сетей

Таблица 6.2.2 Описание строк для файла «actions»

Название свойства	**Формат**	**Описание**
date	Long	Дата создания действия
id	Строка	Идентификатор действия
network	<сеть>	Название сети
type	<тип сущности>	Тип действия

Пример строки файла:

{"date":1430702455000,"id":"vk.com{45045130}group{4285182}post{4290513}comment","network":"vk.com","type":"COMMENT"}

Таблица 6.2.3 Описание строк для файла “create_links”

Название свойства	**Формат**	**Описание**
date	Long	Дата создания связи
dst_id	Строка	Идентификатор действия
network	<сеть>	Название сети
src_id	Строка	Идентификатор автора
type	<тип связи>	Тип действия

Пример строки файла:

{"date":1436409950000,"dst_id":"twitter.com{3269342515}user{619064906512400384}post"
,"network":"twitter.com","src_id":"twitter.com{3269342515}user","type":"CREATE"}

Таблица 6.2.4 Описание строк для файла “follow_links”

Название свойства	**Формат**	**Описание**
date	Long	Дата создания связи
dst_id	Строка	Идентификатор приёмника (на кого происходит воздействие)
network	<сеть>	Название сети
src_id	Строка	Идентификатор источника (кто воздействует)
type	<тип связи>	Тип действия

Пример строки файла:

{"date":1445242052000,"dst_id":"vk.com{137306467}user","network":"vk.com","src_id":"vk.com{15127154}user","type":"FOLLOW"}

Таблица 6.2.5 Описание строк для файла “source_links”

Название свойства	**Формат**	**Описание**
date	Long	Дата создания связи
dst_id	Строка	Идентификатор дочернего действия
network	<сеть>	Название сети
src_id	Строка	Идентификатор родительского действия
type	<тип связи>	Тип действия

Пример строки файла:

{"date":1437028547000,"dst_id":"vk.com{311075755}user{58}post","network":"vk.com","src_id":"vk.com{84875607}group{7161}post","type":"SOURCE"}

Таблица 6.2.6 Описание строк для файла “users”

Название свойства	**Формат**	**Описание**
date	Long	Дата создания действия
id	Строка	Идентификатор действия
name	Строка	Имя пользователя
network	<сеть>	Название сети

Название свойства	**Формат**	**Описание**
type	<тип сущности>	Тип действия
url	Строка	Ардес страницы пользователя в социальной сети

Пример строки файла:

{"date":1444302280000,"id":"vk.com{232307341}user","name":"Данил Герман","network":"vk.com","type":"USER","url":"http://vk.com/d.german2000"}

7. СООБЩЕНИЯ ОПЕРАТОРУ

Обращение к REST API происходит посредством HTTP-запросов. Тело запросов и ответов представлено в JSON формате. Ошибки выполнения функций предоставляются оператору в виде соответствующих кодов состояния HTTP – (англ. HTTP status code) – часть первой строки ответа сервера при запросах по протоколу HTTP. Он представляет собой целое число из трёх десятичных цифр. Первая цифра указывает на класс состояния. За кодом ответа обычно следует отделённая пробелом поясняющая фраза на английском языке, которая разъясняет человеку причину именно такого ответа (см. Таблицу 7.1).

Таблица 7.1 Коды состояния HTTP, используемые для обозначения ошибок

Код	**Описание**	**Рекомендуемые действия для устранения ошибки**
404	«Not Found». Сервер понял запрос, но не нашёл соответствующего ресурса по указанному URL. Относится к группе ошибок со стороны клиента.	Проверить содержание требуемого запроса на правильность запрашиваемого адреса URL и внести соответствующие правки. Либо требуется детальная диагностика системы (анализ логов, инспекция данных и др.).

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

В настоящем документе приняты следующие условные обозначения:

СПО	Специальное программное обеспечение
3i Big Data Analytics Processing Platform	СПО «3i Big Data Analytics Processing Platform»
API	сокр. англ. Application Programming Interface, интерфейс программирования приложений – набор готовых классов, процедур, функций, структур и констант, предоставляемых приложением для использования во внешних программных продуктах
IP-адрес	сокр. англ. Internet Protocol Address – уникальный сетевой адрес узла в компьютерной сети, построенной по протоколу IP
JSON	сокр. англ. JavaScript Object Notation – текстовый формат обмена данными, основанный на JavaScript (происходит от подмножества языка стандарта ECMA-262 1999 года)
Hadoop	Свободно распространяемый набор утилит, библиотек и средство разработки и выполнения распределённых программ, работающих на кластерах из сотен и тысяч узлов.
HDFS	сокр. англ. Hadoop Distributed File System – файловая система, предназначенная для хранения файлов больших размеров, поблочно распределённых между узлами вычислительного кластера

HTTP	сокр. англ. HyperText Transfer Protocol, протокол передачи гипертекста — протокол прикладного уровня передачи данных на технологии «клиент-сервер»
REST API	сокр. англ. Representational State Transfer, передача репрезентативного состояния – метод взаимодействия компонентов распределённого приложения в сети Интернет, при котором вызов удаленной процедуры представляет собой обычный HTTP-запрос

Лист регистрации изменений									
Изм.	Номера листов (страниц)				Всего листов (страниц) в документе	№ доку мента	Входящий № сопроводи тельного документа и дата	Под пись	Дата
	Изме-нен-ных	заменен ных	новых	аннули рован ных					