

1.2 «BIG DATA»

«Big Data» – это та тема, которая активно обсуждается различными компаниями и IT–специалистами. Некоторые из них успели разочароваться в «Big Data», другие – напротив, максимально используют это в различных сферах.

Именно это направление способствовало созданию программных решений для анализа таких как «Hadoop».

Основные пункты, которые рассматриваются в данной главе:

- Определение.
- Сфера применения и сопутствующие технологии.
- Примеры использования в компаниях.
- Выводы.

1.2.1 ОПРЕДЕЛЕНИЕ

Чтобы разобраться в данном вопросе, приведем основные характеристики. «Big Data», на сегодняшний день, являются одним из главных локомотивов развития информационных технологий.

Эта сфера, которая является относительно новым для российского обывателя, получила обширное распространение за рубежом. Это связывается с тем, что в эпоху информационных технологий, особенно после взрывного распространения социальных сетей, по каждому пользователю интернета стало собираться огромное количество информации, что в итоге дало толчок направлению «Big Data».

Термин «Big Data» вызывает множество дискуссий, многие считают, что это означает лишь количество собранной информации, но не стоит забывать и о технической стороне, данное направление включает также технологии хранения, вычисления и сервисные услуги.

Стоит сказать, что к данному направлению относится обработка именно большого количества информации, которое иногда не представляется возможным обработать обычными методами.

На рисунке 6 представлена сравнительная таблица обычной базы и «Big Data».

Характеристика	Традиционная база данных	База Больших Данных
Объем информации	От гигабайт (10^9 байт) до терабайт (10^{12} байт)	От петабайт (10^{15} байт) до эксабайт (10^{18} байт)
Способ хранения	Централизованный	Децентрализованный
Структурированность данных	Структурирована	Полуструктурирована и неструктурирована
Модель хранения и обработки данных	Вертикальная модель	Горизонтальная модель
Взаимосвязь данных	Сильная	Слабая

Источник: Wikibon

Рисунок 6 – сравнительная таблица

Направление «Big Data» имеет следующие признаки:

- «Volume» – объем, накопленная база данных представляет собой большой объем информации, который достаточно трудно обрабатывать и хранить традиционными способами, для них требуются новый подход и усовершенствованные технологии.
- «Velocity» – скорость, данный признак указывает как на увеличивающуюся скорость накопления данных, так и на скорость обработки этих данных, в последнее время стали более востребованы технологии обработки данных в реальном времени.
- «Variety» – многообразие, т.е. возможность одновременной обработки структурированной и неструктуройированной информации разных форматов. Главное отличие структурированной информации – это то, что она может быть классифицирована. Примером такой информации может служить информация о клиентских транзакциях. Неструктурированная информация включает в себя видео, аудио файлы, свободный текст, информацию, поступающую из различных сетей. На сегодняшний день большая часть информации входит в

группу неструктурированной. Данная информация нуждается в комплексном анализе, чтобы сделать ее полезной для дальнейшей обработки.

- «Veracity» – достоверность данных, все большее значение пользователи стали придавать достоверности имеющихся данных. Например, у ИТ-компаний есть проблема по разделению действий, проводимых роботом и человеком, что приводит в конечном счете к затруднению анализа данных.
- «Value» – ценность накопленной информации. «Big Data» должны быть полезны компании и приносить определенную ценность для нее. К примеру, помогать в усовершенствовании бизнес-процессов, составлении отчетности или оптимизации расходов.

При наличии данных признаков, собранные данные можно отнести к числу больших.

1.2.2 СФЕРЫ ПРИМЕНЕНИЯ И СОПУТСТВУЮЩИЕ ТЕХНОЛОГИИ

Сфера применения технологий «Big Data» достаточно обширны. Например, с помощью «Big Data» можно узнать о предпочтениях людей или клиентов, об эффективности тех или иных компаний или провести анализ рисков.

Следует также отметить, что «Big Data» является одной из самых быстроразвивающихся сфер информационных технологий.

Таким образом, «Big Data» – это уже одна из фундаментальных сфер технологий, несмотря на молодой возраст, имеющая распространение во многих сферах бизнеса и играющая важную роль в развитии различных предприятий и компаний.

Основные пункты, которые будут рассматриваться в данной главе:

- Технологии.
- Применение в отраслях.

1.2.2.1 ТЕХНОЛОГИИ

Технологии, используемые в «Big Data» для сбора и обработки информации, можно разделить на три группы:

- Программное обеспечение.
- Оборудование.
- Услуги сервиса.

К наиболее распространенным подходам обработки данных (программному обеспечению) относятся:

- «SQL» – язык структурированных запросов, позволяющий работать с базами данных. С помощью «SQL» можно создавать и модифицировать данные, а управлением массива данных занимается соответствующая система управления базами данных.
- «NoSQL» – термин расшифровывается как «Not Only SQL» (не только «SQL»). Включает в себя ряд подходов, направленных на реализацию базы данных, имеющих отличия от моделей, используемых в традиционных, реляционных СУБД. Их удобно использовать при постоянно меняющейся структуре данных. Например, для сбора и хранения информации в социальных сетях.

- «MapReduce» – модель распределения вычислений. Используется для параллельных вычислений над очень большими наборами данных. В программном интерфейсе код программы передается данным. Таким образом, запрос представляет собой отдельную программу. Подробнее о данной модели будет изложено в следующих главах.
- Экосистема «Hadoop» (которая включает в себя модель «MapReduce»).
- «SAP HANA» – высокопроизводительная «NewSQL» платформа для хранения и обработки данных. Обеспечивает высокую скорость обработки запросов.

К технологическому оборудованию относят:

- Серверы.

Серверы включают в себя хранилища данных.

- Инфраструктурное оборудование.

К инфраструктурному оборудованию относят средства ускорения платформ, источники бесперебойного питания, комплекты серверных консолей и многое другое.

Сервисные услуги включают в себя услуги по построению архитектуры системы базы данных, обустройству, и оптимизации инфраструктуры и обеспечению безопасности хранения данных.

Программное обеспечение, оборудование, а также сервисные услуги вместе образуют комплексные платформы для хранения и анализа данных. Многие компании предлагают услуги по разработке, развертыванию решений «Big Data» и управления ими.

1.2.2.2 ПРИМЕНЕНИЕ В ОТРАСЛЯХ

«Big Data» получили широкое распространение во многих отраслях. Их используют в здравоохранении, телекоммуникациях, торговле, логистике, в финансовых компаниях, а также в государственном управлении.

Примеры применения «Big Data»:

— Розничная торговля.

В базах данных розничных магазинов может быть накоплено множество информации о клиентах, системе управления запасами, поставками товарной продукции. Данная информация может быть полезна во всех сферах деятельности магазинов.

Например, с помощью собранной информации можно управлять поставками товара, его хранением и продажей. На основании информации можно прогнозировать спрос и поставки товара. Также система обработки и анализа данных может решить и другие проблемы, например, оптимизировать затраты или подготовить отчетность.

— Финансовые услуги.

«Big Data» дает возможность проанализировать кредитоспособность заемщика, также они полезны для кредитного скоринга и андеррайтинга. Внедрение технологий «Big Data» позволит сократить время рассмотрения кредитных заявок. С помощью «Big Data» можно проанализировать операции конкретного клиента и предложить для него персонализированные услуги.

— Телекоммуникации.

В телекоммуникационной отрасли широкое распространение «Big Data» получила у сотовых операторов.

Операторы сотовой связи наравне с финансовыми организациями имеют одни из самых объемных баз данных, что позволяет им проводить наиболее глубокий анализ накопленной информации.

Основной задачей анализа данных является удержание существующих клиентов и привлечение новых. Для этого компании проводят сегментацию клиентов, анализируют их трафики, определяют социальную принадлежность абонента.

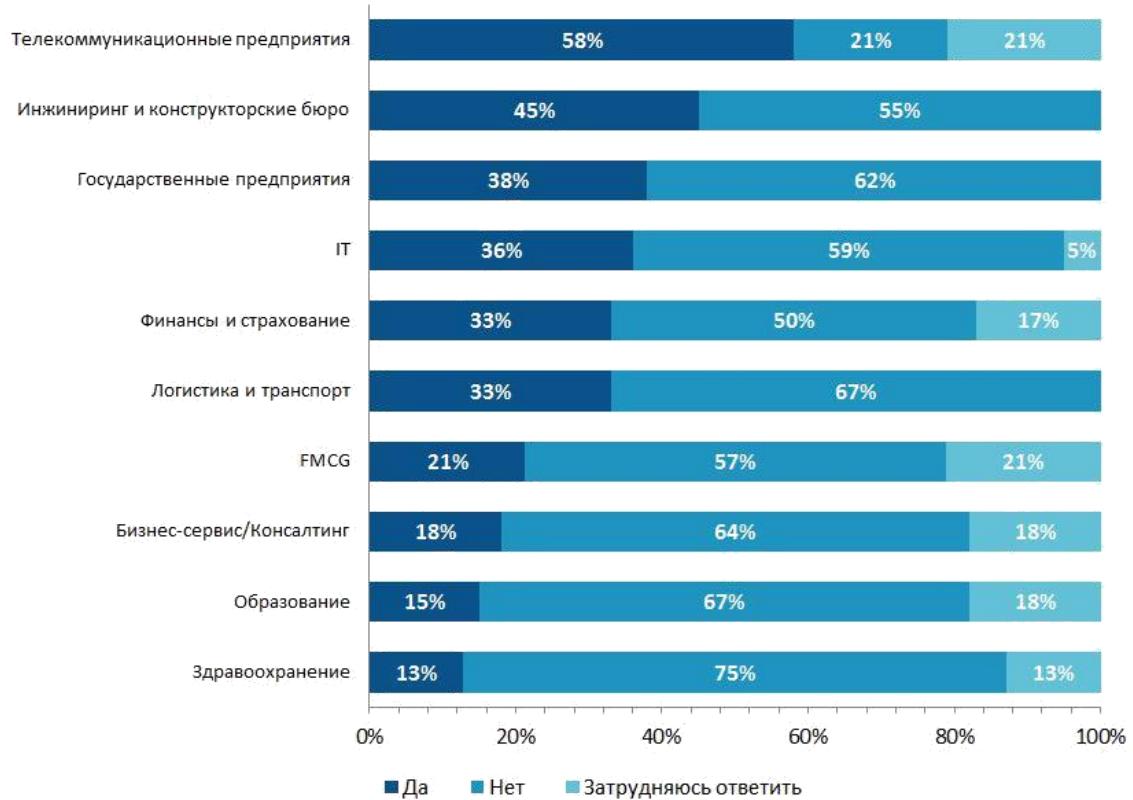
— Горнодобывающая и нефтяная промышленности.

«Big Data» используется как при добыче полезных ископаемых, так и при их переработке и сбыте. Предприятия могут на основании доступной информации делать выводы об эффективности разработки месторождений, прогнозировать спрос на продукцию и цены.

По данным, представленным на рисунке 7, наибольшее распространение «Big Data» получила в телекоммуникационной отрасли, а также в инжиниринге, ИТ, в финансовых и государственных предприятиях.

Также можно заметить очень маленький спрос в сферах здравоохранения, образования и консалтинга.

Компании из каких отраслей внедрили технологии Больших Данных?



Источник: Tech Pro Research

Рисунок 7 – использование «Big Data» в различных отраслях

1.2.3 ПРИМЕРЫ ИСПОЛЬЗОВАНИЯ В КОМПАНИЯХ

На сегодняшний день «Big Data» активно внедряется в зарубежных компаниях. Такие компании, как «Facebook», «Google», «IBM», «VISA», «Master Card», «Bank of America», «AT&T», «Coca Cola» и «Starbucks» уже используют ресурсы «Big Data».

Сфера применения обработанной информации разнообразны и изменяются в зависимости от отрасли и задач, которые необходимо решать.

Некоторые примеры применения технологий «Big Data» на практике:

— «HSBC» использует технологии «Big Data» для противодействия мошеннических операций с картами. С помощью «Big Data» компания увеличила эффективность службы безопасности в три раза, распознавание мошеннических инцидентов – в десять раз.

— Антифрод «VISA» позволяет в автоматическом режиме вычислить операции мошеннического характера. Система на данный момент помогает предотвратить мошеннические платежи на сумму два миллиарда долларов ежегодно.

— Суперкомпьютер «Watson» компании «IBM» анализирует в реальном времени поток данных по денежным транзакциям. По данным «IBM», «Watson» на пятнадцать процентов увеличил количество обнаруженных мошеннических операций, на пятьдесят процентов сократил ложные срабатывания системы.

— С помощью «Yandex Data Factory» разработчики игры «World of Tanks» анализируют поведение игроков. Технологии «Big Data» позволили проанализировать поведение ста тысяч игроков «World of Tanks» с использованием более ста параметров. В результате анализа был получен прогноз ухода пользователей. Эта информация позволяет уменьшить уход пользователей и работать с участниками игры адресно. Разработанная модель оказалась на двадцать пять процентов эффективнее стандартных инструментов анализа игровой индустрии.

— Министерство труда Германии использует «Big Data» в работе, связанной с анализом поступающих заявлений на выдачу пособий по безработице. Проанализировав информацию, было выявлено, что двадцать процентов пособий выплачивалось незаслуженно. С помощью «Big Data» министерство труда сократило расходы на десять миллиардов евро.

1.2.4 ВЫВОДЫ

В эпоху расцвета информационных технологий «Big Data» является очень важным направлением на сегодняшний момент, так как требуется хранить и обрабатывать огромное количество информации, проанализировав которое, компании в различных сферах могут принимать взвешенные решения в вопросах развития бизнеса.

1.3 ЭКОСИСТЕМА «HADOOP»

Как уже говорилось ранее, экосистема «Hadoop» является одной из важнейших технологий «Big Data».

Основные пункты, которые рассматриваются в данной главе:

- Описание платформы.
- Компоненты экосистемы «Hadoop».
- Описание основных компонентов.
- Выводы.

1.3.1 ОПИСАНИЕ ПЛАТФОРМЫ

«Hadoop» – это программная платформа с открытым исходным кодом, находящийся под управлением «Apache Software Foundation». «Hadoop» используется для надежных, масштабируемых и распределенных вычислений, так как обладает большой вычислительной мощью, но может также применяться и как хранилище данных, которое может вместить гигантское количество информации.

Данная платформа широко распространена во многих сферах, где имеет место работа с «Big Data».

Вокруг «Hadoop» образовалась целая экосистема из связанных проектов и технологий, многие из которых с самого начала развивались в рамках проекта, а позже перешли в самостоятельные.

Сейчас идет активный процесс коммерциализации технологий, несколько компаний строят бизнес, который полностью направлен на создание коммерческих дистрибутивов «Hadoop» и услуг по поддержке экосистемы.

Стоит отметить, что практически все крупные компании, специализирующиеся на поставке информационных технологий для организаций, включают «Hadoop» (в том или ином виде) в свои линейки решений.

Основные пункты, которые будут рассматриваться в данной главе:

- История создания.
- Использование и преимущества.

1.3.1.1 ИСТОРИЯ СОЗДАНИЯ

Разработка данной системы была инициирована в начале 2005 года, с целью построения программной инфраструктуры распределённых вычислений для проекта «Nutch» – свободного поисковика на «Java», её идейной основой стала публикация «Google» о вычислительной концепции «MapReduce».

Данный проект был назван в честь игрушечного слонёнка, который принадлежал ребенку основателя проекта.

В течение времени «Hadoop» развивался, сначала в рамках проекта «Nutch», затем – проекта «Lucene». В 2008 году «Yahoo» выпускал «Hadoop», как проект с открытым исходным кодом.

Сегодня, платформой «Hadoop» и экосистемой технологий распоряжается некоммерческая организация «Apache Software Foundation» (ASF), глобальное сообщество разработчиков программного обеспечения.

1.3.1.2 ИСПОЛЬЗОВАНИЕ И ПРЕИМУЩЕСТВА

Экосистема «Hadoop» используется в следующих компаниях:

- «IBM».
- «ebaY».
- «Twitter».
- «Mail.ru group».
- «Amazon».
- «Adobe».
- «Яндекс».
- «Yahoo».
- «Facebook».
- «LinkedIn».

Особенности данной платформы:

- Работает с «Big Data» на обычных серверах.
- Сильное открытое сообщество.
- Множество различных продуктов и средств используют «Hadoop».

Для развертывание кластера «Hadoop» обычно используют обычные сервера (не суперкомпьютеры или десктопы), соединенные по сети и расположенные в одном месте (сервера в стойках в «датацентре»).

Преимущества «Hadoop»:

- Возможность хранить и обрабатывать огромное количество данных.

С развитием социальных сетей и прочих решений, приходиться иметь дело с постоянно растущими объемами данных. Поэтому это очень важный фактор.

- Вычислительная мощность.

Модель распределенных вычислений «Hadoop» обрабатывает большие данные быстро. Чем больше вычислительных узлов, которые вы используете, тем больше вычислительной мощности вы имеете.

- Отказоустойчивость.

Данные и обработка приложений защищены от отказа оборудования. Если узел теряет работоспособность, то задания автоматически перенаправляются к другим узлам, чтобы удостовериться, что распределенные вычисления не перестали работать. Многократные копии (репликация) всех данных производятся автоматически.

- Гибкость.

В отличие от традиционных реляционных баз данных, вы не должны предварительно обрабатывать данные прежде, чем сохранить их. Вы можете хранить столько данных, сколько вы хотите (разных типов) и решаете, как использовать их позже. Это включает неструктурированные данные как текст, изображения и видео.

- Низкая стоимость.

Платформа с открытым исходным кодом бесплатна и использует товарные аппаратные средства, чтобы хранить большое количество данных.

— Масштабируемость (горизонтальная).

Вы можете легко нарастить свою систему, чтобы обработать больше данных, просто добавив узлы. И для этого не требуется глубокого администрирования.

— Инкапсуляция сложности реализации.

«Hadoop» скрывает многие сложности распределенных и многопоточных систем. Освобождает разработчика от заботы о проблемах системного уровня, таких как ожидание данных, организация передачи данных, распределение данных, доставка кода. Позволяет разработчику сфокусироваться на разработке приложений и реализации бизнес–логики.

1.3.2 КОМПОНЕНТЫ ЭКОСИСТЕМЫ «HADOOP»

Изначально, «Hadoop» был инструментом для хранения данных и запуска «MapReduce» задач, сейчас «Hadoop» представляет собой большой сборник технологий, так или иначе связанных с обработкой больших данных.

Основные пункты, которые будут рассматриваться в данной главе:

- Описание компонентов.
- Дистрибутивы экосистемы «Hadoop».

1.3.2.1 ОПИСАНИЕ КОМПОНЕНТОВ

Основными компонентами (ядро) «Hadoop» являются:

- «Hadoop Distributed File System» (HDFS) – распределённая файловая система, позволяющая хранить информацию практически неограниченного объёма.
- «YARN» – программная модель для управления ресурсами и менеджмента задач, в том числе включает программную модель «MapReduce» (используются в новых версиях «Hadoop» вместо «MapReduce»).
- «Hadoop common» – библиотеки и утилиты, которые используются другими модулями «Hadoop».
- «HadoopMapReduce» – программный каркас для программирования распределённых вычислений в рамках парадигмы «MapReduce».

На рисунке 8 показана схема основных компонентов экосистемы «Hadoop»:

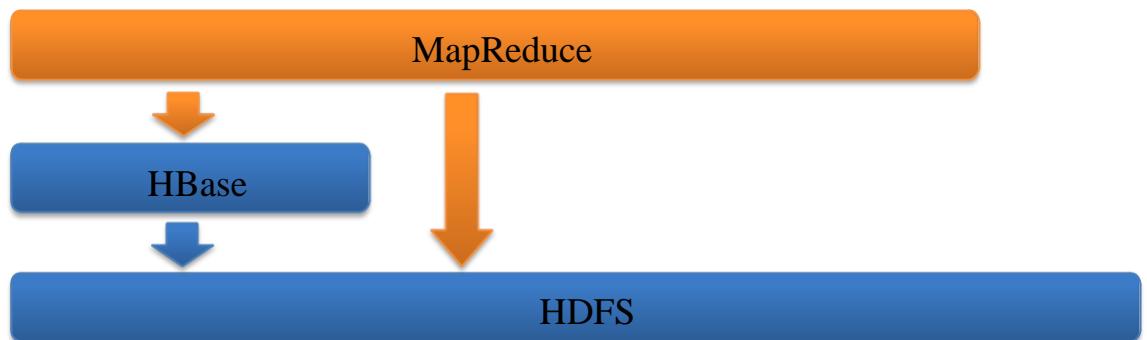


Рисунок 8 – схема основных компонентов экосистемы «Hadoop»

Также существует большое количество проектов, непосредственно связанных с «Hadoop», но который не входят в набор основных компонентов «Hadoop»:

- «Hive» – инструмент для «SQL-like запросов» над большими данными (превращает «SQL-запросы» в серию «MapReduce-задач»).
- «Pig» – язык программирования для анализа данных на высоком уровне. Одна строчка кода на этом языке может превратиться в последовательность «MapReduce-задач».
- «HBase» – колоночная база данных, реализующая парадигму «BigTable».
- «Cassandra» – высокопроизводительная распределенная «key-value» база данных.
- «ZooKeeper» – сервис для распределённого хранения конфигураций и синхронизации изменений этих конфигураций.
- «Mahout» – библиотека и движок машинного обучения на больших данных.

На рисунке 9 показана схема всех компонентов экосистемы «Hadoop»:

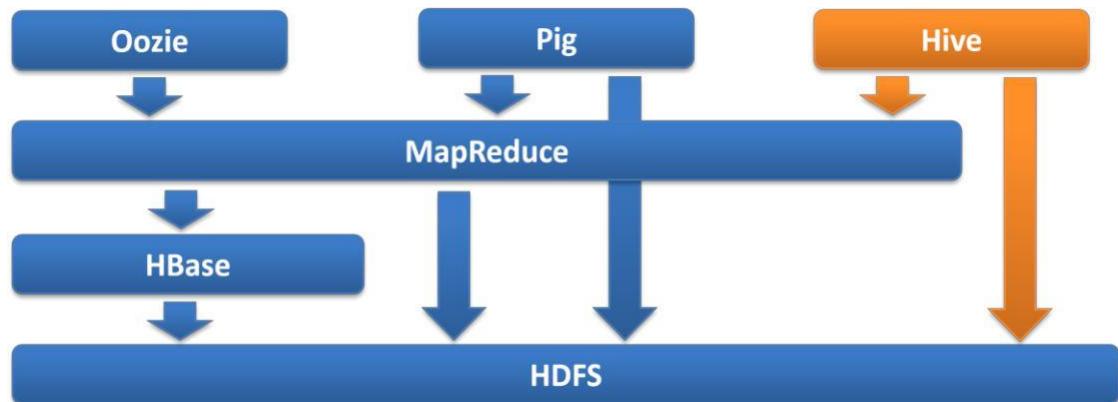


Рисунок 9 – схема всех компонентов экосистемы «Hadoop»

Данные модули расширяют возможности платформы, что позволяет разработчикам хранить и анализировать различные типы данных.

1.3.2.2 ДИСТРИБУТИВЫ ЭКОСИСТЕМЫ «HADOOP»

При установке «Hadoop» пользователь сталкивается с несовместимостью версий различных компонентов «Hadoop» (так как их надо устанавливать отдельно). Эта проблема решается путем выбора решений от сторонних поставщиков.

Поставщики дистрибутивов «Hadoop» обеспечивают:

- Интеграционные тесты различных компонентов «Hadoop».
- Установочные пакеты в разных форматах.
- Некоторые поставщики добавляют дополнительные функции и исправляют ошибки в стандартной версии «Hadoop».

Основные поставщики экосистемы «Hadoop»:

- «Cloudera Distribution for Hadoop» (CDH).
- «MapR Distribution».
- «Hortonworks Data Platform» (HDP).
- «Apache BigTop Distribution».
- «Greenplum HD Data Computing Appliance».

В данной работе мы будем использовать дистрибутив от «Cloudera».

1.3.3 ОПИСАНИЕ ОСНОВНЫХ КОМПОНЕНТОВ

Основные пункты, которые будут рассматриваться в данной главе:

- Распределенная файловая система «HDFS».
- Программная модель «MapReduce».

1.3.3.1 РАСПРЕДЕЛЕННАЯ ФАЙЛОВАЯ СИСТЕМА «HDFS»

Экспоненциальный рост информации привел к потребности в появлении файловых систем, ориентированных на обеспечение высокой производительности, масштабируемости, надежности и доступности.

Так появилась распределенная файловая система «HDFS» – «Hadoop Distributed File System» (аналог «GSF» от «Google»). Проект активно поддерживается и развивается компанией «Yahoo».

Основные пункты, которые будут рассматриваться в данной главе:

- Определение и составные части.
- Хранение и репликация данных.
- Запись и удаление данных.
- Преимущества и недостатки.

1.3.3.1.1 ОПРЕДЕЛЕНИЕ И СОСТАВНЫЕ ЧАСТИ

«HDFS» – распределенная файловая система, используемая в проекте «Hadoop». «HDFS–кластер» в первую очередь состоит из следующих компонентов:

- «NameNode» – основной сервер для управления пространством имен файловой системы и доступом клиентов к данным. Данный сервер, при первом запуске, фиксирует все транзакции, связанные с изменением метаданных файловой системы, в файле под названием «EditLog» и применяет данные изменения к образу «HDFS», расположенный в файле «FsImage». Затем записывается уже новый образ с изменениями, и система начинает работу с чистым «EditLog».
- «DataNode» – сервер для непосредственного хранения данных. Передача данных происходит непосредственно между «DataNode» и клиентом, чтобы разгрузить «NameNode».
- «Secondary NameNode» – сервер, который выполняет функции «NameNode», касающийся файлов «EditLog» и «FsImage». При этом данные файлы хранятся на основном сервере.

1.3.3.1.2 ХРАНЕНИЕ И РЕПЛИКАЦИЯ ДАННЫХ

Данные в «HDFS» хранятся в виде блоков (блок – единица хранения файлов) на «DataNode» и управляетяется через «NameNode». Причем при записи файла происходит процесс репликации (копирования) данного файла.

Стандартный размер блоков в «HDFS» – 64МБ или 128 МБ. Основным мотивом для определения такого размера является уменьшение времени поиска

данных по сравнению с их передачей. При этом происходит процесс репликации (копирования) данных.

Копии блоков хранятся на нескольких серверах, по умолчанию – трех. Их размещение происходит следующим образом:

- Первая реплика размещается на локальном «DataNode».
- Вторая реплика на другом «DataNode» в этой же стойке.
- Третья реплика на произвольной «DataNode» другой стойки.
- Остальные реплики размещаются произвольным способом.

На рисунке 10 изображена схема хранения данных в «HDFS»:

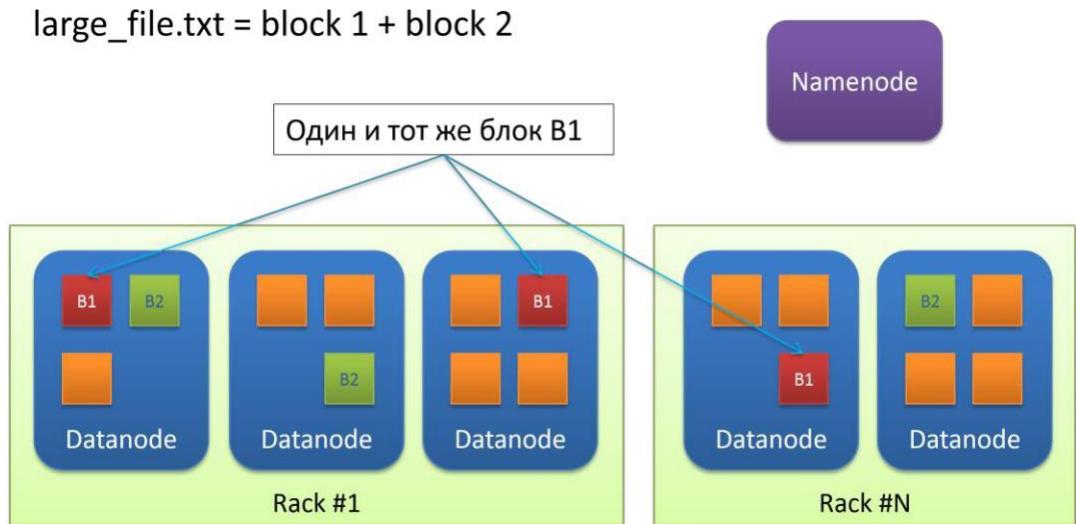


Рисунок 10 – схема хранения данных в «HDFS»

При чтении данных клиент выбирает ближайший к нему «DataNode» с репликой.

1.3.3.1.3 ЗАПИСЬ И УДАЛЕНИЕ ДАННЫХ

При записи данных в «HDFS» используется подход, позволяющий достигнуть высокой пропускной способности.

Приложение ведет запись в потоковом режиме, при этом «HDFS-клиент» кэширует записываемые данные во временном локальном файле. Когда в файле накапливаются данные на один блок, клиент обращается к «NameNode», который регистрирует новый файл, выделяет блок и возвращает клиенту список «DataNode» для хранения реплик блока.

Клиент начинает передачу данных блока из временного файла первому «DataNode» из списка. «DataNode» сохраняет данные на диске и пересыпает следующему «DataNode».

Таким образом, данные передаются в конвейерном режиме и копируются на требуемом количестве серверов. По окончании записи, клиент уведомляет «NameNode», который фиксирует транзакцию создания файла, после чего он становится доступным в системе.

В силу обеспечения сохранности данных (на случай отката операции), удаление в файловой системе происходит по определенной методике.

Вначале файл перемещается в специально отведенную для этого «/trash» директорию, а уже после истечения определенного времени, происходит его физическое удаление:

- Удаление файла из пространства имен «HDFS».
- Освобождение связанных с данными блоков.

1.3.3.1.4 ПРЕИМУЩЕСТВА И НЕДОСТАТКИ

Текущие недостатки:

- Отсутствие автоматического запуска главного сервера в случае его сбоя (данная функциональность реализована в GFS).
- «NameNode» – является самым слабым звеном в системе. В случае его сбоя, упадет вся система.
- Нет многопоточной записи. Один процесс записи на файл. Данные дописываются в конец файла.
- Не подходит для большого количества маленьких файлов.

Преимущества:

- Хранение огромного количества данных.
- Оптимизация под последовательное чтение. Высокая пропускная способность.
- Простота. Воспринимается пользователем как «один большой жесткий диск».
- Не требует дорогостоящего оборудования.

1.3.3.2 ПРОГРАММНАЯ МОДЕЛЬ «MAPREDUCE»

Основные пункты, которые будут рассматриваться в данной главе:

- Определение и описание работы модели.
- Архитектура «Hadoop MapReduce».
- Преимущества и недостатки.

1.3.3.2.1 ОПРЕДЕЛЕНИЕ И ОПИСАНИЕ РАБОТЫ МОДЕЛИ

«MapReduce» – программная модель для выполнения распределенных вычислений для больших объемов данных, представляющая собой набор «Java-классов» и исполняемых утилит для создания и обработки заданий на параллельную обработку.

Основные принципы «MapReduce» можно сформулировать так:

- Обработка и вычисление больших объемов данных.
- Масштабируемость.
- Автоматическое распараллеливание заданий.
- Работа на простом оборудовании.
- Автоматическая обработка отказов выполнения заданий.

Работу «MapReduce» можно поделить на следующие этапы:

- Чтение входных данных.

Входные данные делятся на блоки данных предопределенного размера (от 16 Мб до 128 Мб), которые называются сплиты. «MapReduce» закрепляет за каждой функцией «Мар» определенный сплит.

- Функция «Мар».

Каждая функция Мар получает на вход список пар «ключ/значение», обрабатывает их и на выходе получает ноль или более пар «ключ’/значение’», являющихся промежуточным результатом.

Все операции «Мар» выполняются параллельно и не зависят от результатов работы друг друга. Каждая функция «Мар» получает на вход свой уникальный набор данных.

- Функция «Reduce».

Программная модель вызывает функцию «Reduce» для каждого уникального ключа в списке значений. Операции «Reduce» выполняются параллельно и не зависят от результатов работы друг друга.

Таким образом, результаты работы каждой функции «Reduce» пишутся в отдельный блок «HDFS».

— Запись выходных значений.

Результаты, полученные на этапе «Reduce», записываются в файловые блоки в «HDFS». Каждый узел «Reduce» пишет в собственный блок.

На рисунке 11 отображена схема работы «MapReduce»:

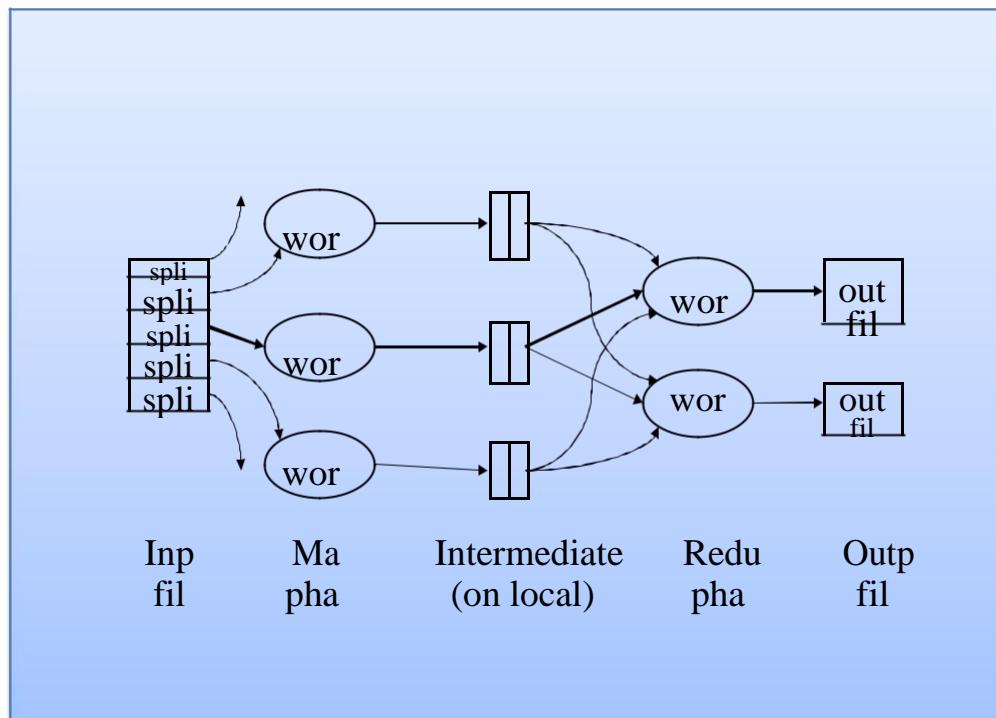


Рисунок 11 – схема работы «MapReduce»

Разработчику приложений для «Hadoop MapReduce» необходимо написать базовый обработчик, который на каждом вычислительном узле обеспечит образование начальных пар «ключ/значение» в промежуточный набор пар «ключ'/значение'» (класс, реализующий интерфейс «Mapper»), и обработчик,

сводящий промежуточный набор пар в окончательный набор пар (класс, реализующий интерфейс «Reducer»).

Все остальные этапы выполняются программной моделью без дополнительных усилий разработчика.

«MapReduce» выполняет следующие функции:

- Планирование заданий.
- Распараллеливание заданий.
- Перенос заданий к данным.
- Синхронизация выполнения заданий.
- Обработка отказов выполнения заданий и их перезапуск.
- Оптимизация сетевых взаимодействий.

1.3.3.2.2 АРХИТЕКТУРА «HADOOP MAPREDUCE»

«MapReduce» использует архитектуру «Master–Worker», где «Master» – единственный экземпляр управляющего процесса («JobTracker»), запущенный на отдельной машине. «Worker» – это произвольное множество процессов «TaskTracker», исполняющихся на «DataNode».

«JobTracker» и «TaskTracker» находятся над уровнем хранения «HDFS», и запускаются по следующим правилам:

- Экземпляр «JobTracker» исполняется на «NameNode».
- Экземпляры TaskTracker исполняются на «DataNode».

Вышеописанные принципы расположения «JobTracker» и «TaskTracker» позволяют существенно сократить объемы передаваемых по сети данных и сетевые задержки, связанные с передачей этих данных.

«JobTracker» является единственным узлом, на котором выполняется приложение «MapReduce», вызываемое клиентом.

«JobTracker» выполняет следующие функции:

- Планирование индивидуальных (по отношению к «DataNode») задач «Мар» и «Reduce».
- Координация задач.
- Мониторинг выполнения задач.
- Переназначение провалившихся задач другим узлам «TaskTracker».

В свою очередь, «TaskTracker» выполняет следующие функции:

- Выполнение задач «Мар» и «Reduce».
- Управление исполнением задач.
- Отправка сообщений о статусе задачи узлу «JobTracker».
- Отправка диагностических сообщений узлу «JobTracker».

Взаимодействие «TaskTracker» с «JobTracker» идет посредством «RPC-вызовов», причем они идут только от «TaskTracker». Такое решение уменьшает зависимость управляющего процесса «JobTracker» от процессов «TaskTracker».

Взаимодействие «JobTracker» с клиентом проходит по следующей схеме:

- «JobTracker» принимает задание от клиента и разбивает задание на множество «Мар» задач и множество «Reduce» задач. Узел «JobTracker» использует информацию о файловых блоках, расположенную в «NameNode», чтобы решить, сколько заданий нужно создать на узлах «TaskTracker».
- «TaskTracker» получает от «JobTracker» список задач, загружает код и выполняет его. С определенной периодичностью «TaskTracker» отсылает «JobTracker» статус о выполнении задач.

Взаимодействия «TaskTracker» с клиентом отсутствуют.

При сбое «TaskTracker» «JobTracker» переназначает задания другому узлу «TaskTracker».

В случае неисправности «JobTracker», для продолжения исполнения «MapReduce–задачи», необходим перезапуск «JobTracker». При перезапуске «JobTracker» считывает из журнала данные, о последней успешной контрольной точке, восстанавливает свое состояние на момент записи и продолжает работу.

1.3.3.2.3 ПРЕИМУЩЕСТВА И НЕДОСТАТКИ

Основные преимущества:

- Эффективная работа с большим объемом данных.
- Масштабируемость.
- Отказоустойчивость.
- Открытый исходный код.

Недостатки «MapReduce»:

- Эффективность применение «MapReduce» снижается при малом количестве компьютеров.
- Нельзя определить окончание стадии «Мар».
- Задержки в исполнении любой запущенной «Мар» задачи ведет к задержке выполнения задачи целиком.
- Сбой узла «JobTracker» приводит к простою всего кластера.

1.3.4 ВЫВОДЫ

Данная платформа предоставляет обширные возможности по анализу и хранению данных любого размера и типа, благодаря большому количеству различных компонентов и решений, а также активной поддержке сообщества разработчиков данной платформы.

2 РУКОВОДСТВО ПО УСТАНОВКЕ И ТЕСТИРОВАНИЮ «HADOOP»

В данной работе, практическое руководство будет разделено на два этапа:

- Установка экосистемы «Hadoop» на персональный компьютер.
- Написание программы «WordCount» для тестирования системы «Hadoop».

2.1 УСТАНОВКА ЭКОСИСТЕМЫ «HADOOP» НА ПЕРСОНАЛЬНЫЙ КОМПЬЮТЕР

В данной главе будет представлено пошаговое руководство по установке экосистемы «Hadoop» на персональный компьютер под управлением операционной системы «Windows 10».

Так как «Hadoop» работает на операционных системах «Linux», нам понадобится развернуть на компьютере виртуальную машину, которая будет работать под управлением данной системы.

Прежде чем приступить к установке, скачайте необходимые файлы:

- Программу для виртуализации операционных систем «Oracle VM VirtualBox»: <https://www.virtualbox.org/> .
- Операционную систему «Linux Ubuntu 16.04.2 LTS»:
<http://ubuntu.ru/get> (разрядность нужной системы «Linux» определяется по разрядности нашей текущей операционной системы).

- Дистрибутив «Hadoop» от «Cloudera»:
<https://archive.cloudera.com/cdh5/cdh/5/hadoop-2.6.0-cdh5.11.0.tar.gz> (на установленной системе «Linux»).

Руководство по установке экосистемы «Hadoop» будет разделено на три этапа:

- Установка программы для виртуализации операционных систем.
- Установка операционной системы.
- Установка экосистемы «Hadoop».

2.1.1 УСТАНОВКА ПРОГРАММЫ ДЛЯ ВИРТУАЛИЗАЦИИ ОПЕРАЦИОННЫХ СИСТЕМ

После того, как вы скачали файл программы «Oracle VM VirtualBox» с официального сайта, запустите его.

Появится окно установки как на рисунке 12. Нажмите кнопку «Next».



Рисунок 12 – окно установки программы

Оставьте стандартные параметры установки как на рисунках 13 и 14.
Нажимайте кнопку «Next».

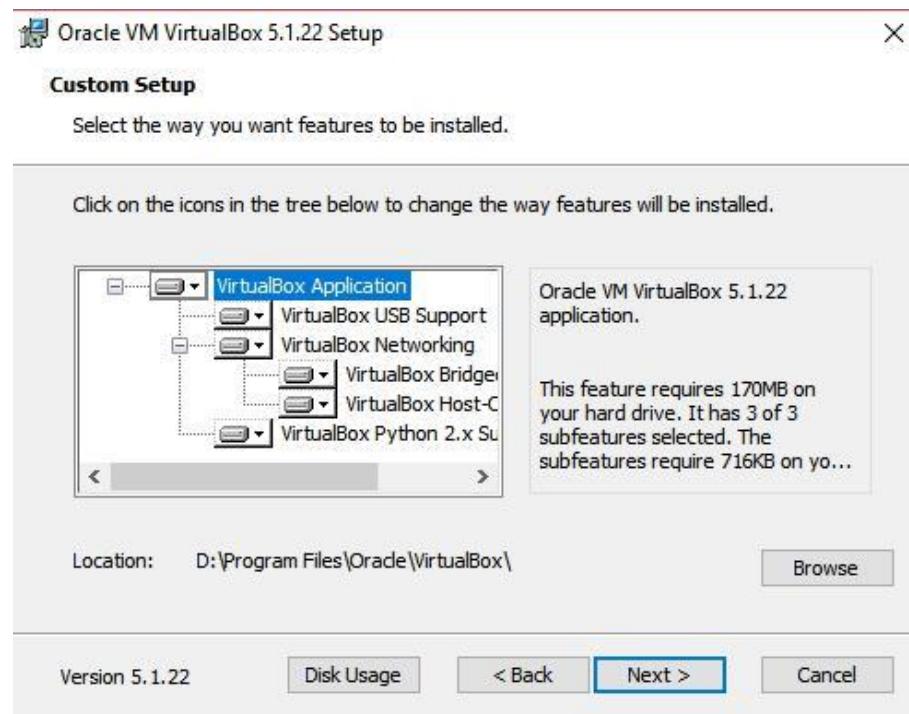


Рисунок 13 – параметры установки

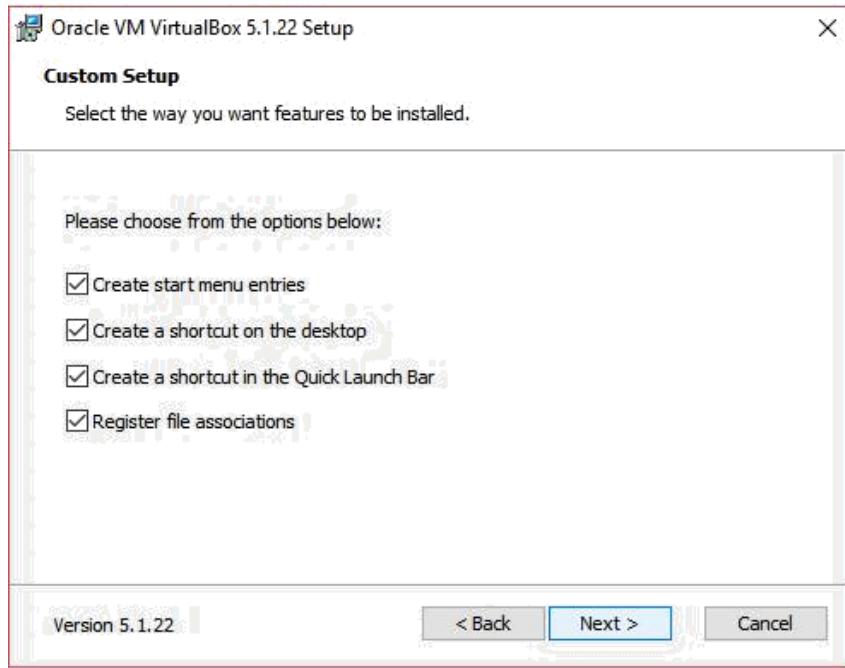


Рисунок 14 – параметры установки

После окончания предварительных настроек, начните установку программы, нажав на кнопку «Install» (рисунок 15).

Дождитесь завершения установки и запустите программу, отметив соответствующий параметр (рисунок 16).

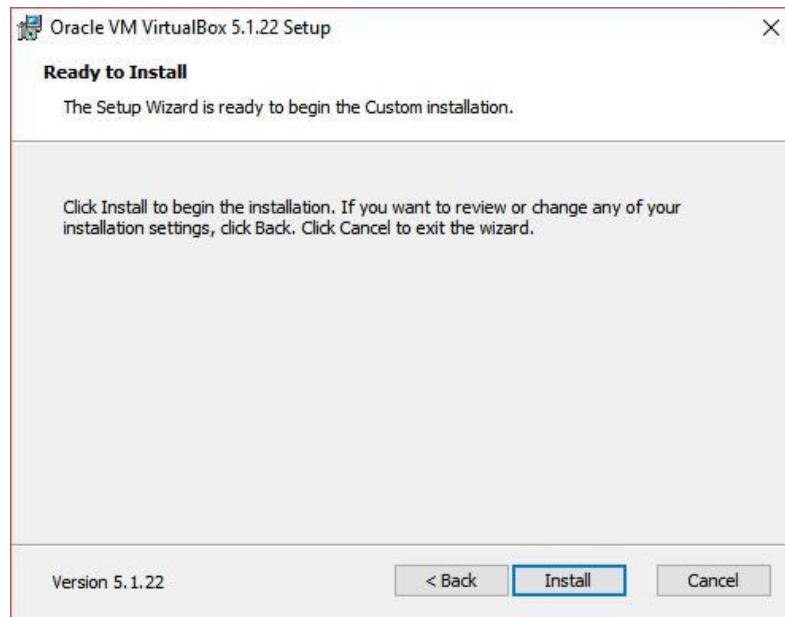


Рисунок 15 – окно начала установки



Рисунок 16 – окно завершения установки

Откроется окно программы как на рисунке 17. Установка программы для виртуализации операционных систем завершено.

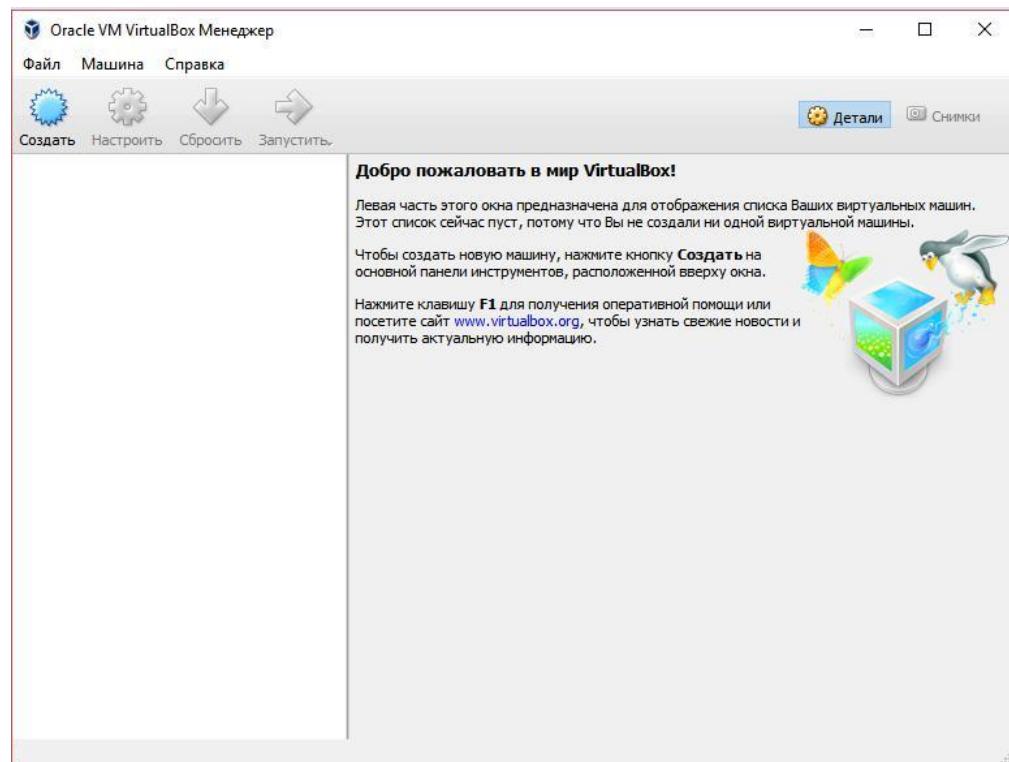


Рисунок 17 – окно программы

2.1.2 УСТАНОВКА ОПЕРАЦИОННОЙ СИСТЕМЫ

Как уже говорилось ранее, «Hadoop» работает на системах «Linux».

Приступим к установке «Linux Ubuntu 16.04.2 LTS».

Перейдите в окно программы «VirtualBox», нажмите кнопку «Создать» на панели управления.

Появится окно создания виртуальной машины как на рисунке 18. Введите следующие параметры и нажмите кнопку «Next».

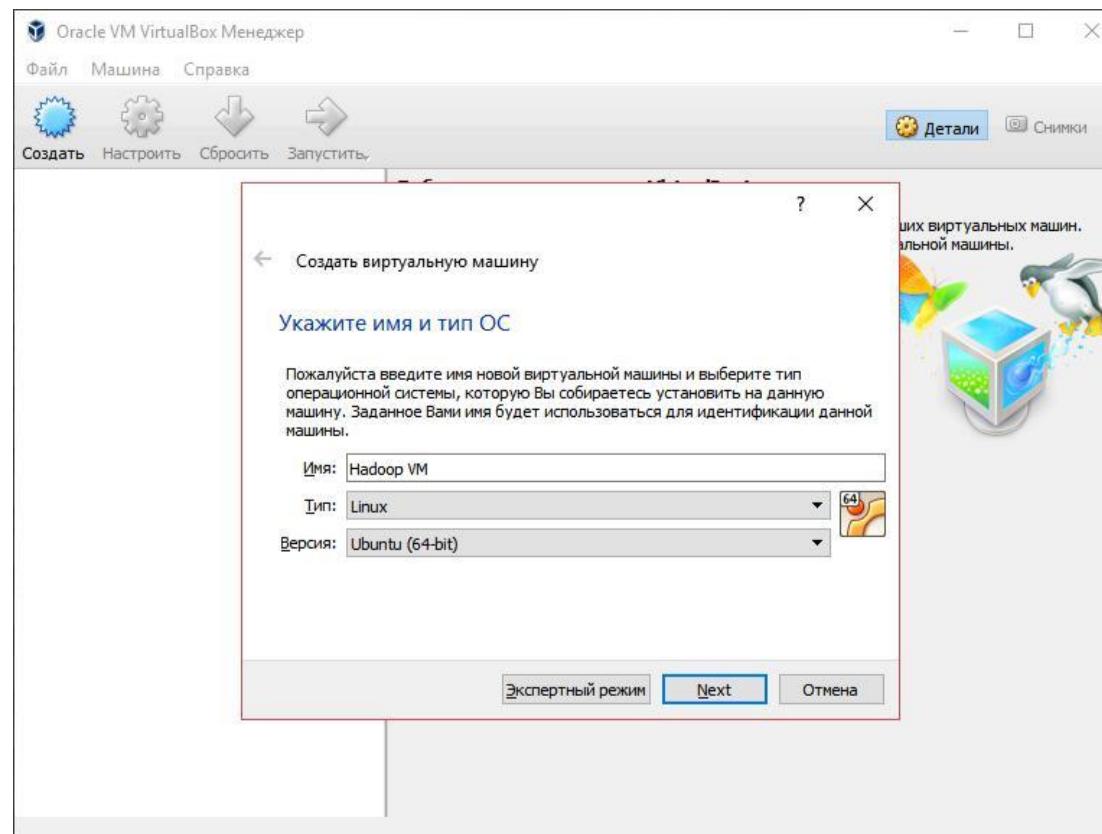


Рисунок 18 – окно создания виртуальной машины

Укажите объем оперативной памяти не менее 2ГБ, нажмите кнопку «Next» (рисунок 19).

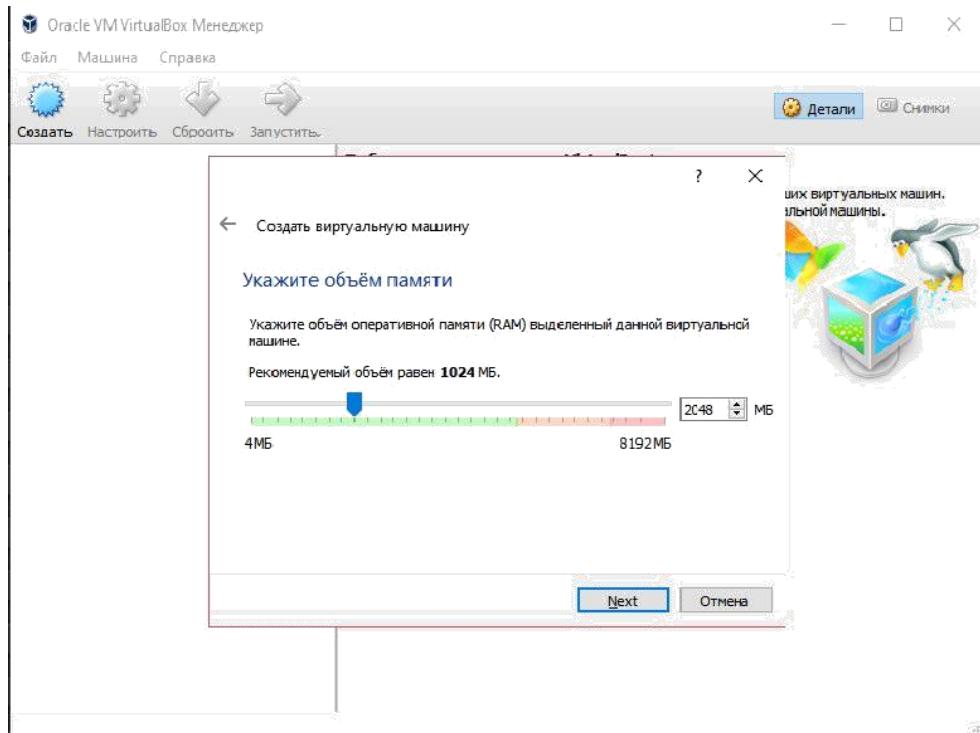


Рисунок 19 – указание объема оперативной памяти

В окне создания жесткого диска отметьте пункт «Создать новый виртуальный жесткий диск», нажмите кнопку «Создать» (рисунок 20).

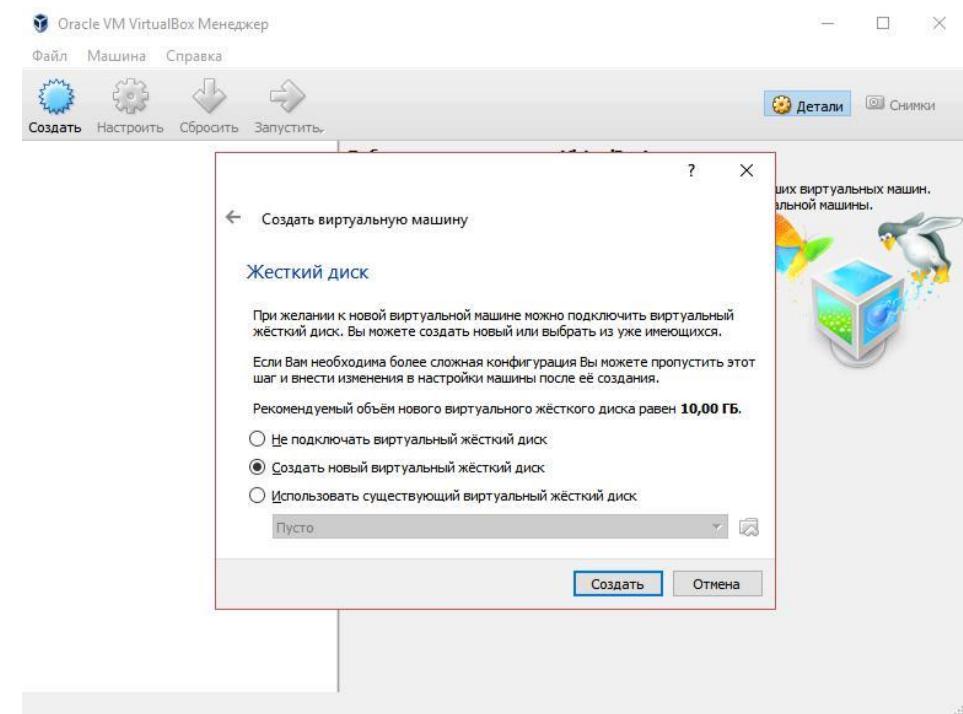


Рисунок 20 – окно создания жесткого диска

В качестве типа укажите «VDI (Virtual Disk Image)» (рисунок 21).

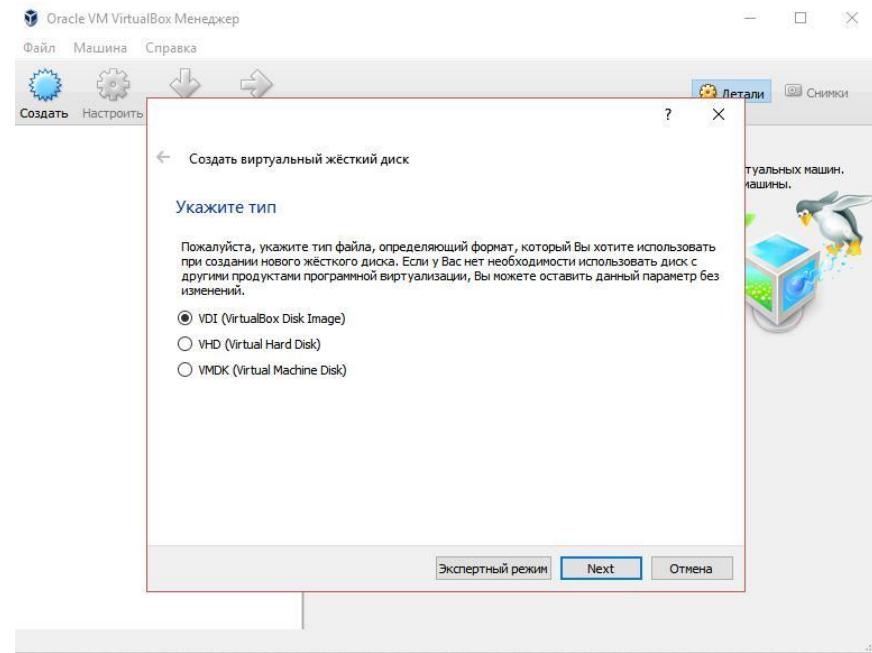


Рисунок 21 – указание типа жесткого диска

В качестве формата хранения укажите «Динамический виртуальный жесткий диск» (рисунок 22).

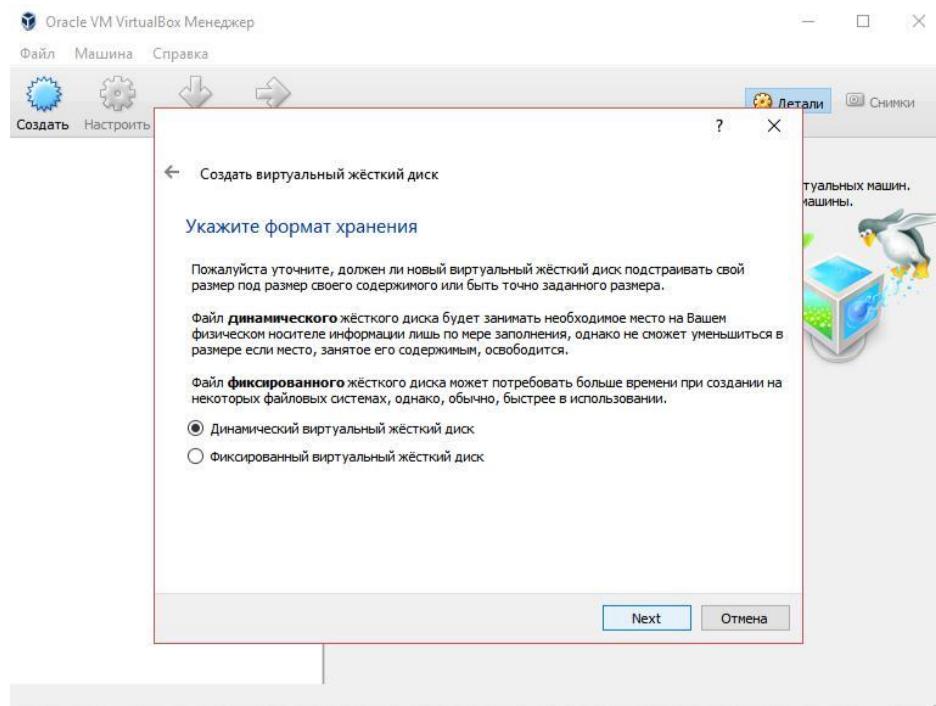


Рисунок 22 – указание формата жесткого диска

Оставьте после с именем нового виртуального жесткого диска нетронутым.
Установите размер жесткого диска не менее 10 ГБ (рисунок 23).

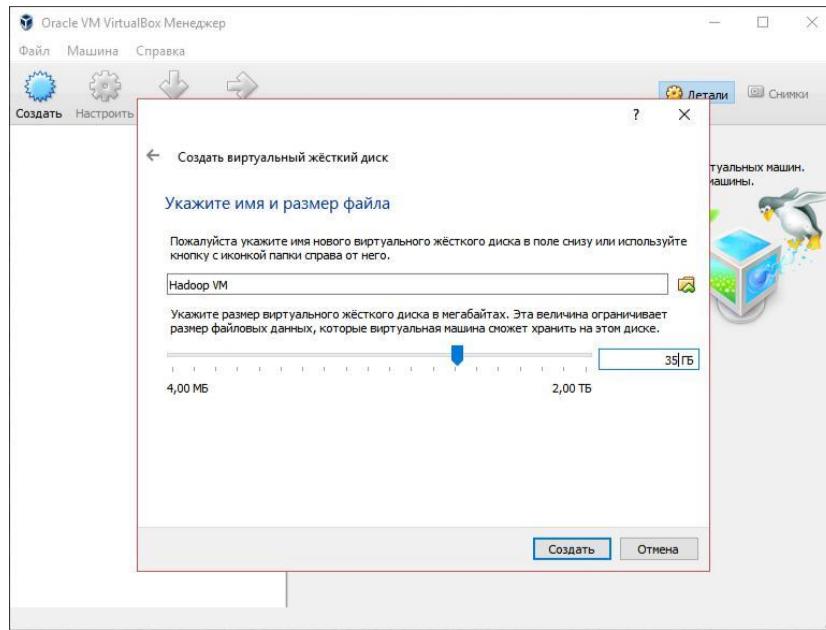


Рисунок 23 – установка размера жесткого диска

После всех, проделанных выше, шагов по созданию и настройке, в окне программы «VirtualBox» появится наша виртуальная машина (рисунок 24).

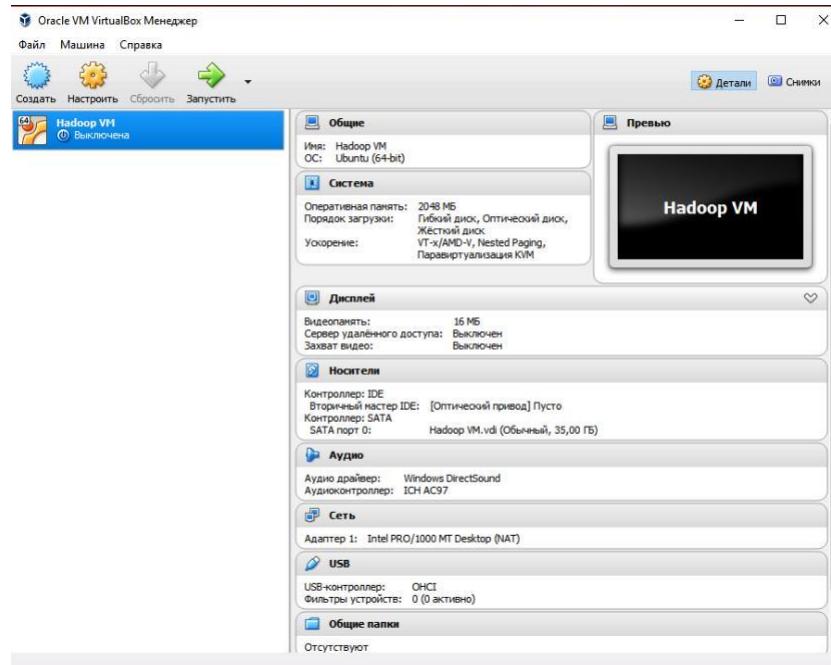


Рисунок 24 – окно программы с виртуальной машиной

Выделите данную виртуальную машину и нажмите кнопку «Настроить» на панели управления.

Перейдите в настройку «Дисплей», на вкладке «Экран» отметьте параметр «Включить 3D–ускорение». Нажмите «OK» (рисунок 25).

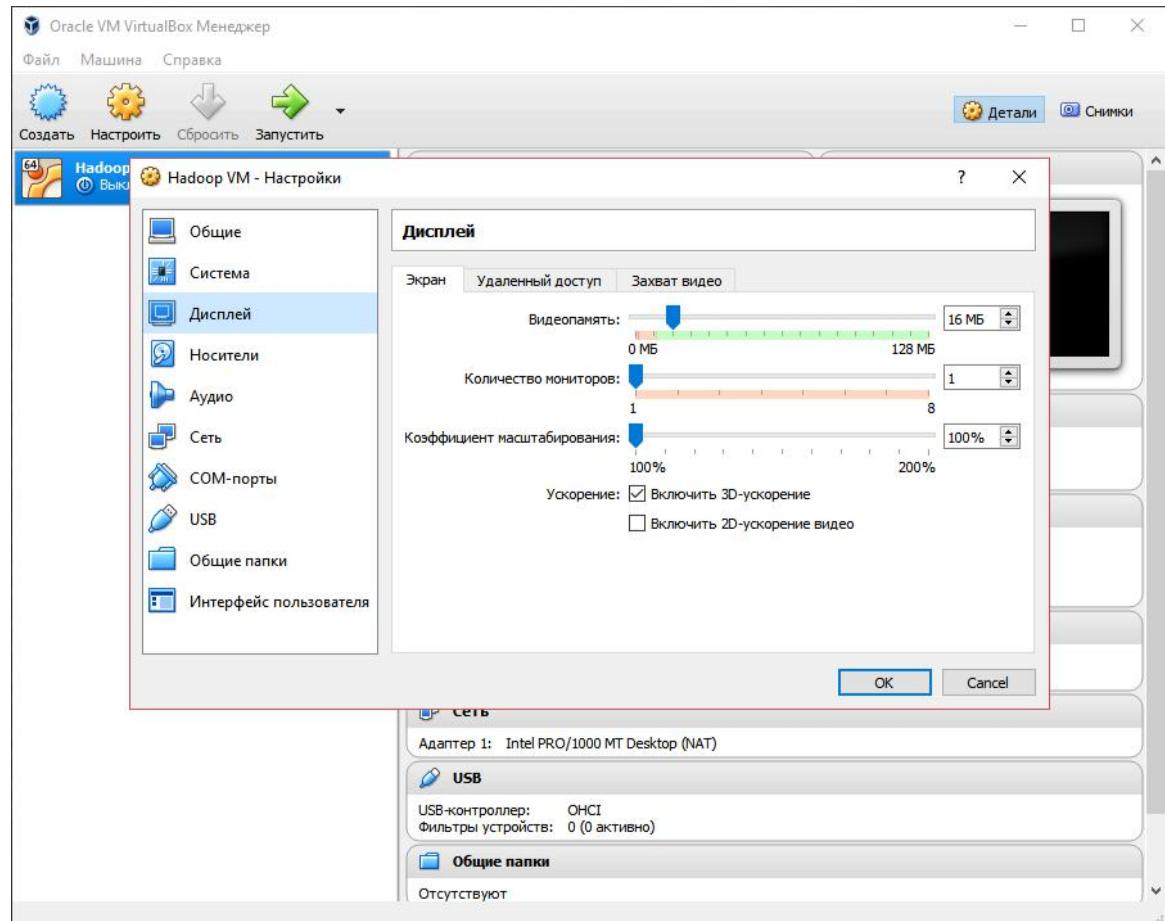


Рисунок 25 – настройка виртуальной машины

Теперь мы можем приступить к установке операционной системы на нашу виртуальную машину.

Нажмите кнопку «Запустить» на панели управления. Машина запуститься и потребует загрузочный диск (который мы скачивали ранее).

Укажите путь к данному диску и нажмите кнопку «Продолжить» (рисунок 26).

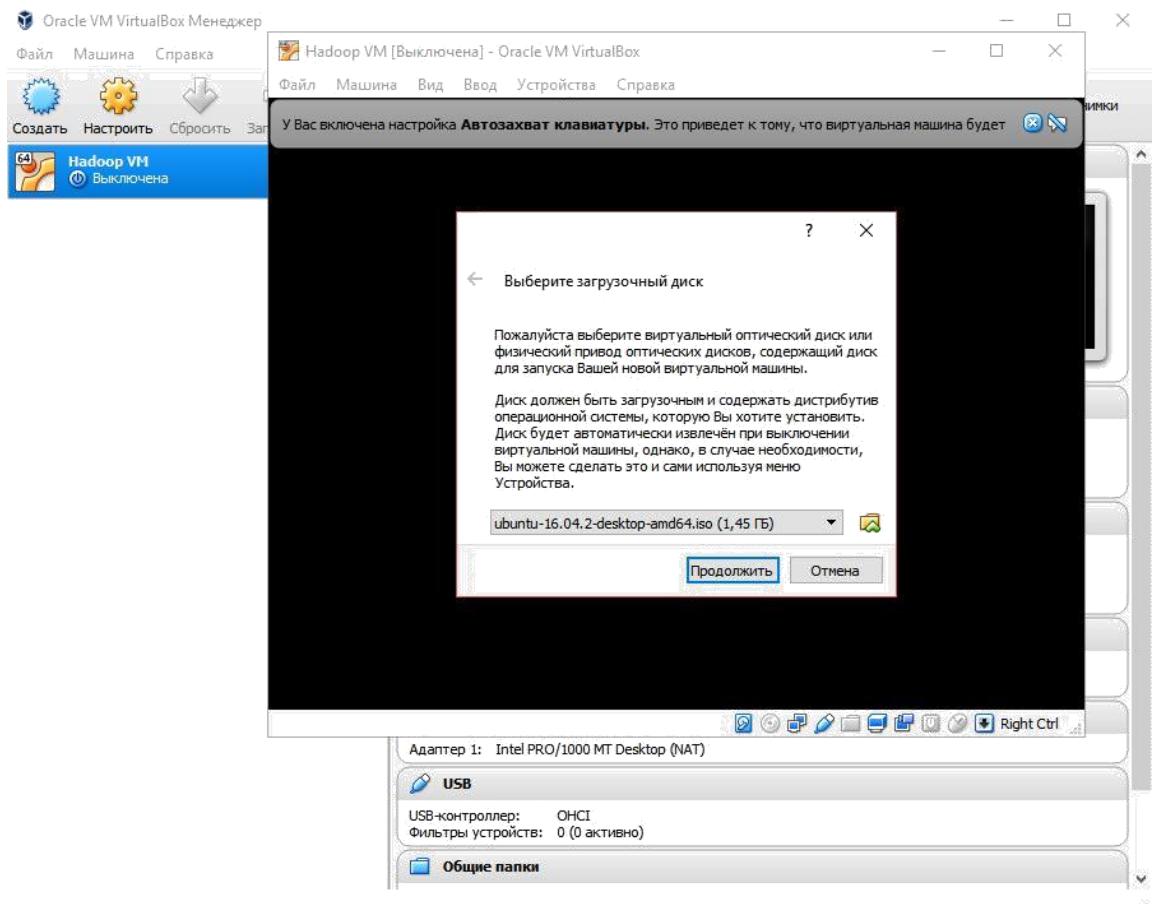


Рисунок 26 – выбор загрузочного диска

После загрузки, появится окно установки операционной системы «Linux». Выберите русский язык установки и нажмите кнопку «Установить Ubuntu» (рисунок 27).

Во время установки потребуется ввести следующие параметры:

- Имя – «HadoopUser».
- Имя компьютера – «HadoopVM».
- Имя пользователя – «hadoopuser».
- Пароль – «thispass123».

Так же для удобства следует отметить автоматический вход в систему (без ввода пароля).

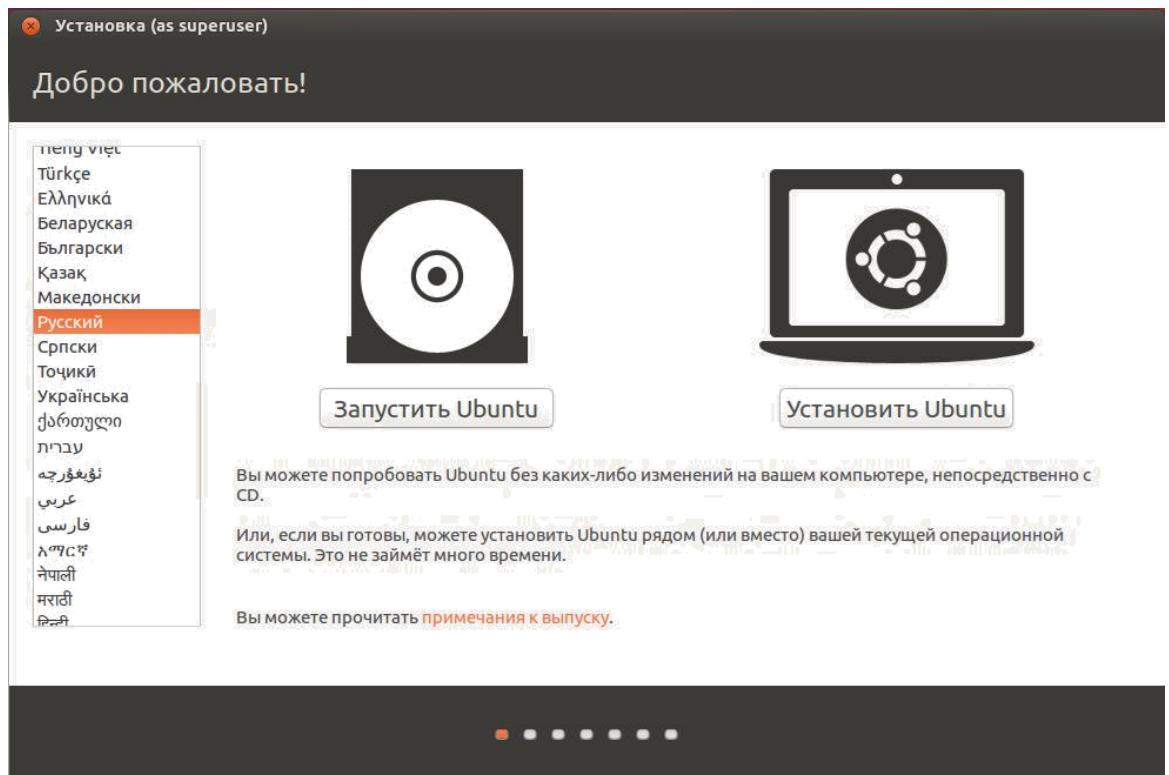


Рисунок 27 – окно установки операционной системы

Отметьте параметры «Запустить обновления во время установки Ubuntu» и «Установить стороннее программное обеспечение для видеокарт и устройств Wi-Fi, а также Flash, MP3 и других медиаданных».

Нажмите кнопку «Продолжить» (рисунок 28).

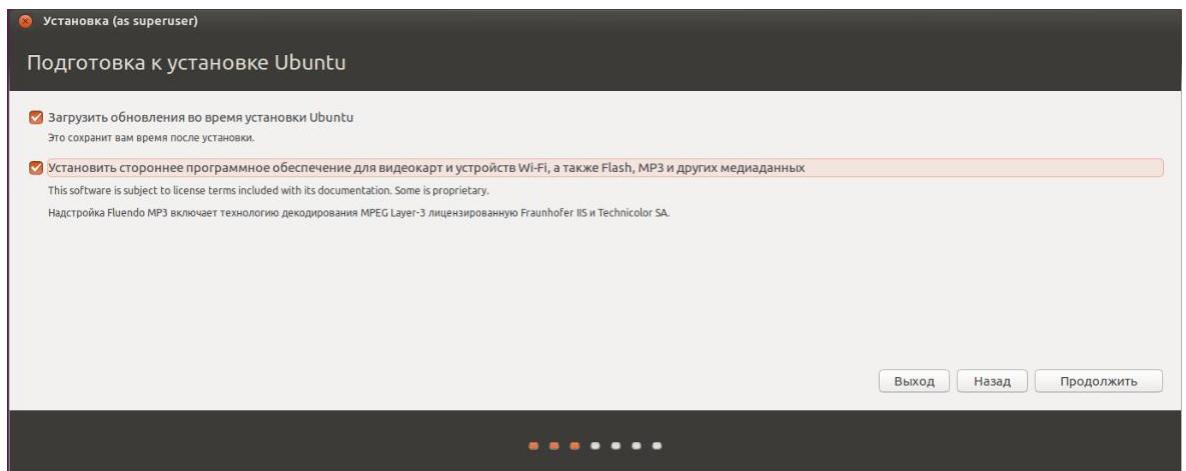


Рисунок 28 – подготовка к установке

В окне «Тип Установки» указанным должен быть только один параметр – «Стереть диск и установить Ubuntu».

Нажмите «Установить сейчас» (рисунок 29).

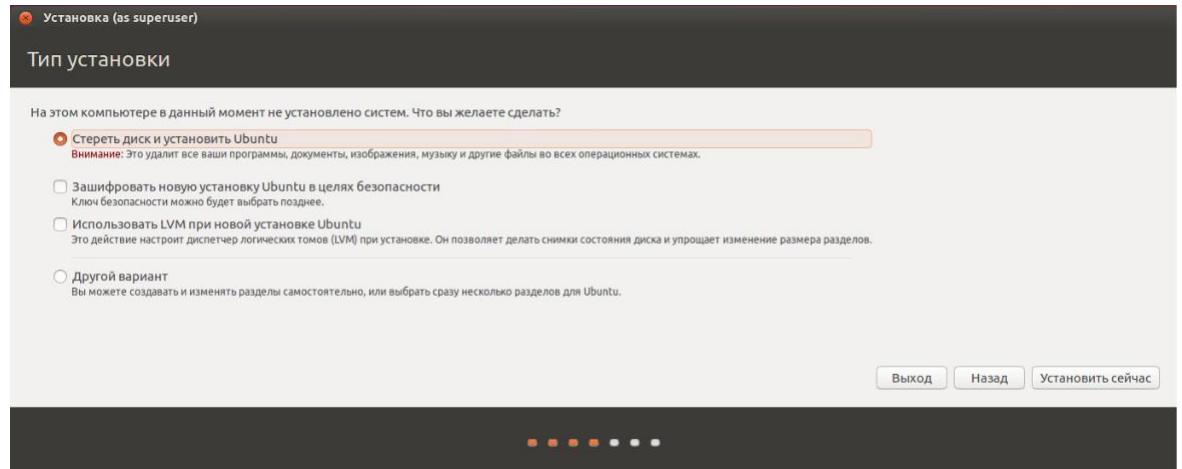


Рисунок 29 – настройка установки

При возникновении предупреждения (рисунок 30), нажмите кнопку «Продолжить».

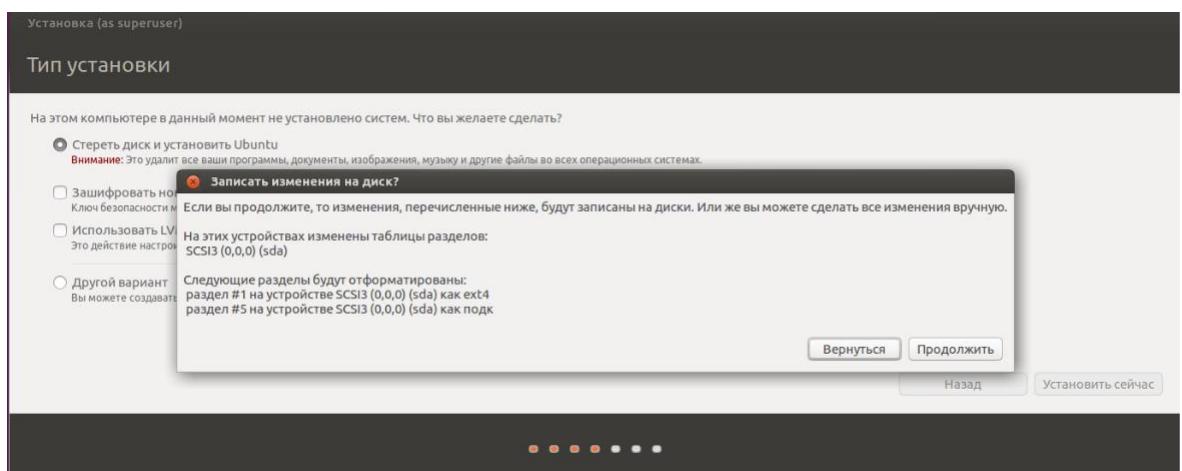


Рисунок 30 – окно предупреждения

Установите русскую раскладку, нажмите кнопку «Продолжить» (рисунок 31).

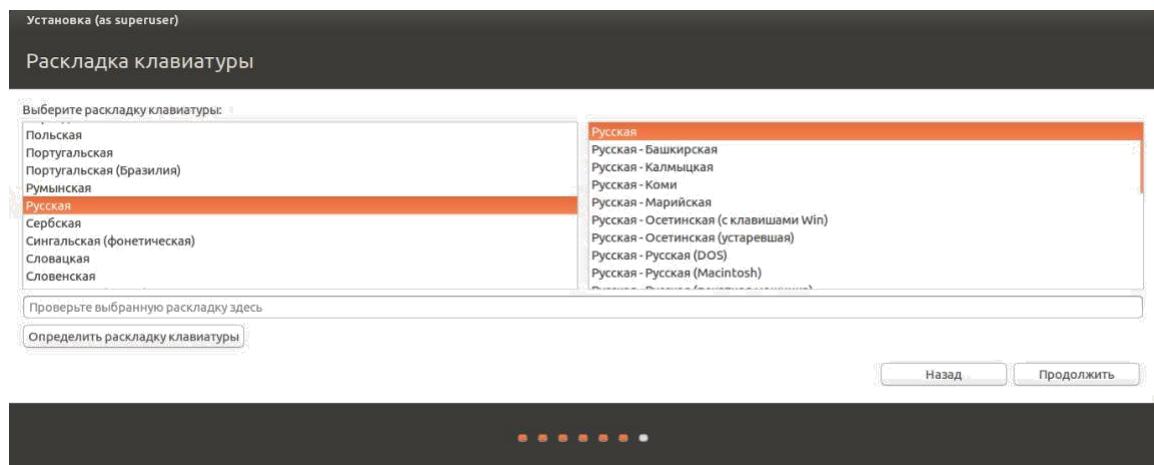


Рисунок 31 – установка раскладки операционной системы

Дождитесь окончания установки (рисунок 32).

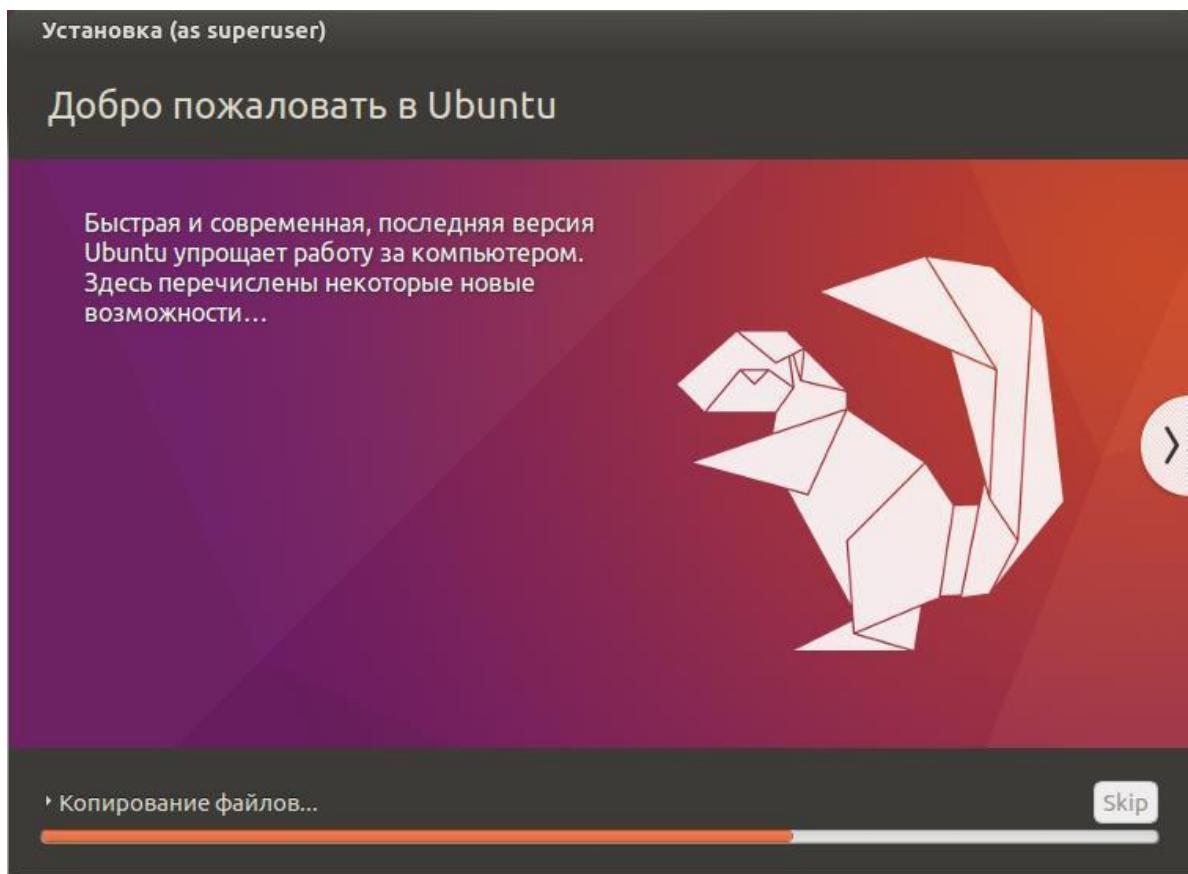


Рисунок 32 – процесс установки операционной системы

После завершения установки, перезагрузите систему, нажав соответствующую кнопку «Перезагрузить» (рисунок 33).

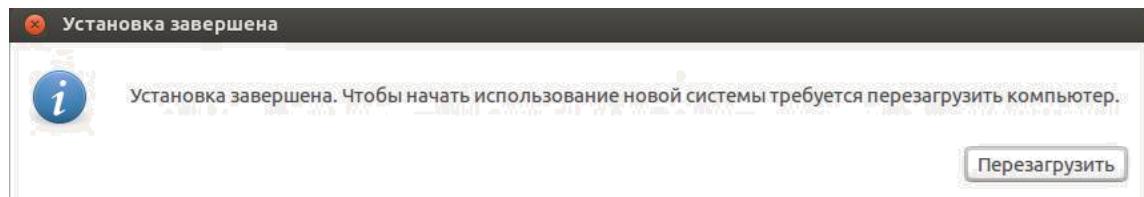


Рисунок 33 – окно завершения установки

На следующем запуске виртуальной машины нам потребуется извлечь установочный диск.

Для этого выберите «Устройства – Оптические диски – Изъять диск из привода». Затем нажмите клавишу «Enter» (рисунок 34).

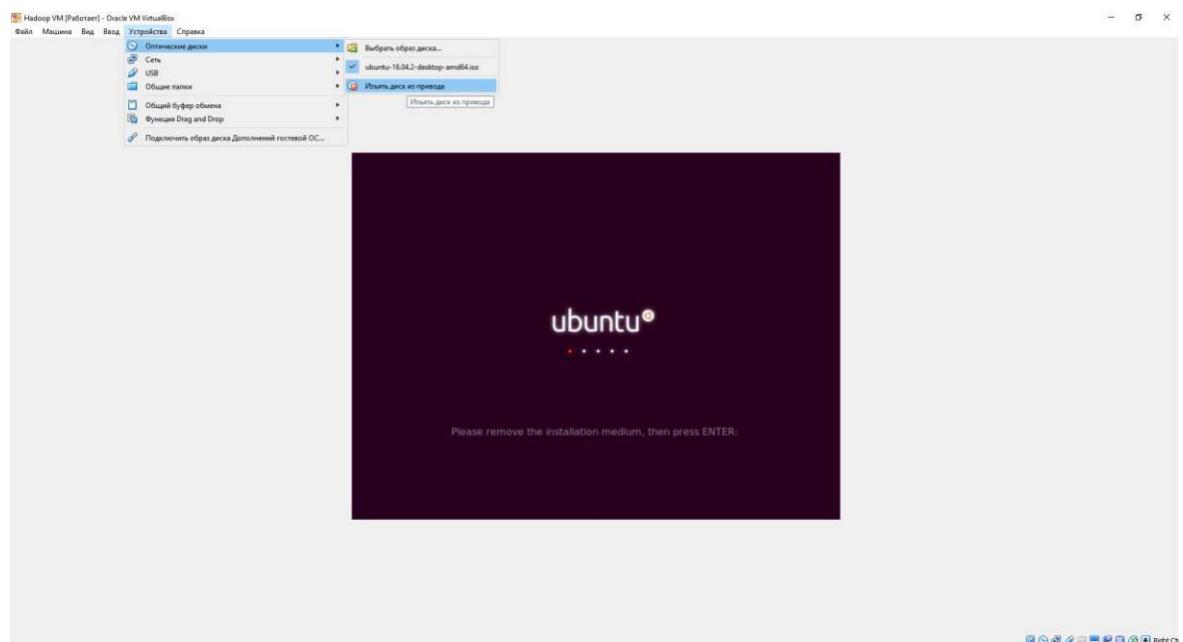


Рисунок 34 – извлечение установочного диска

Установка операционной системы завершена.

Теперь потребуется установить дополнения для комфортной работы с нашей виртуальной машиной.

Для этого выберите «Устройства – Подключить образ диска Дополнений гостевой ОС» (рисунок 35).

На возникшем диалоговом окне нажмите кнопку «Запустить» (рисунок 36).

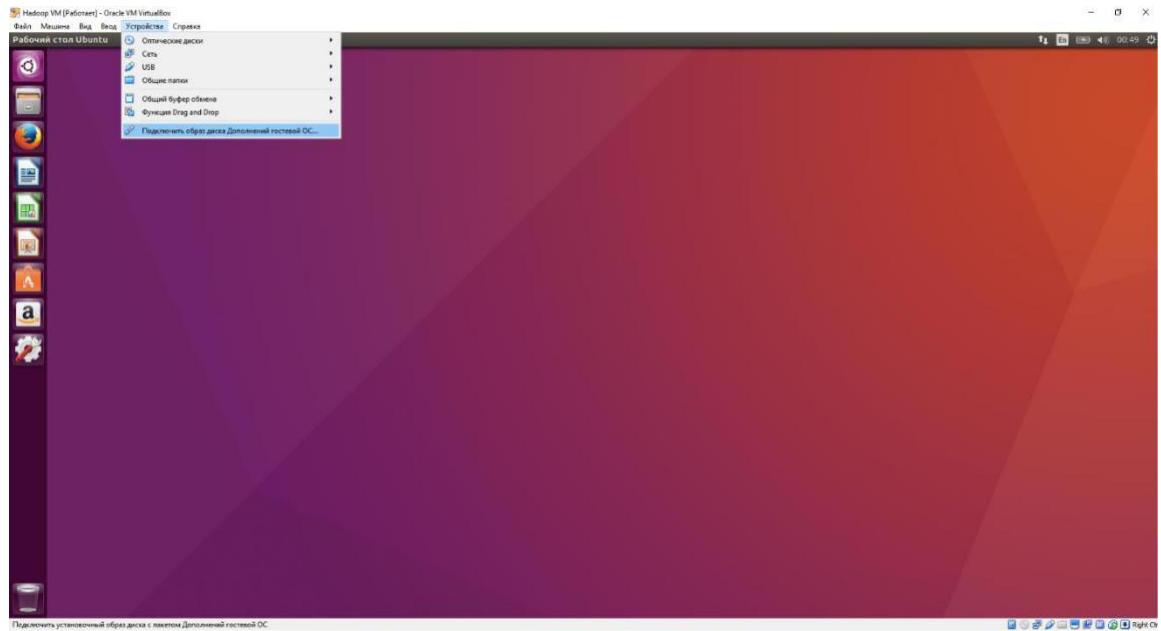


Рисунок 35 – подключение диска дополнений

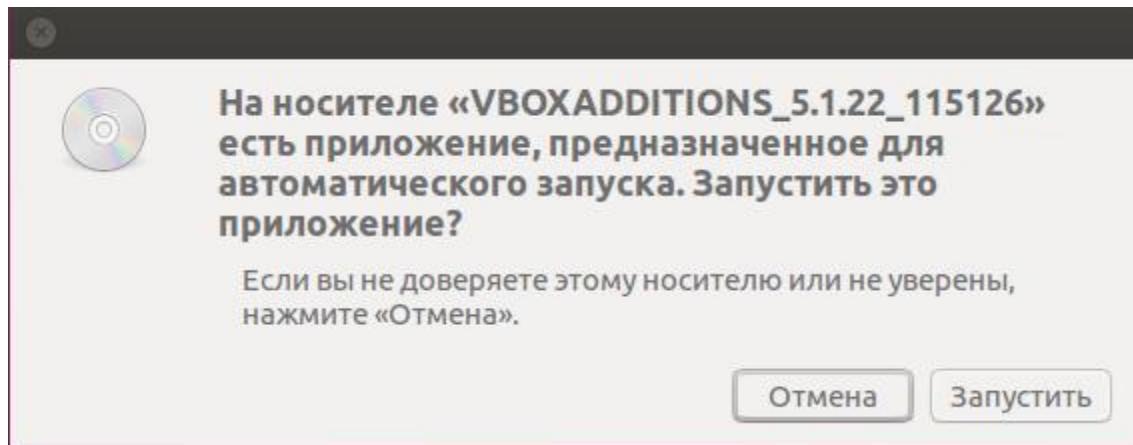


Рисунок 36 – окно запуска установки дополнений

Возникнет терминал и окно аутентификации. В данном окне введите пароль пользователя (thispass123) и нажмите кнопку «Аутентифицировать» (рисунок 37).

Дождитесь выполнения и закрытия терминала (рисунок 38).

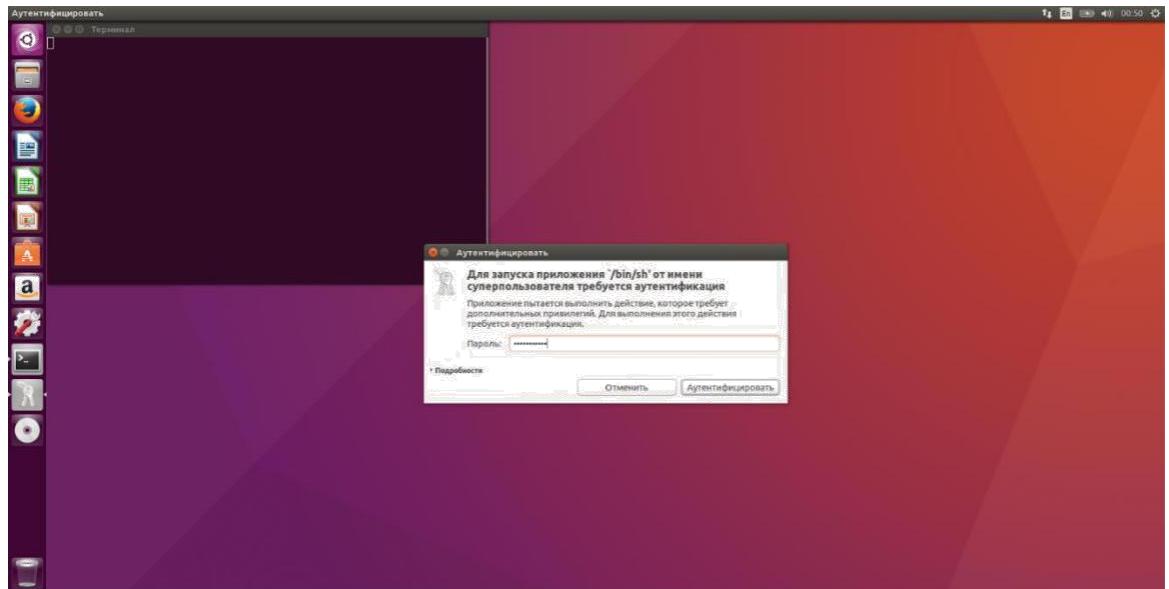


Рисунок 37 – окно аутентификации

```
Verifying archive integrity... All good.
Uncompressing VirtualBox 5.1.22 Guest Additions for Linux.....
VirtualBox Guest Additions installer
Copying additional installer modules ...
Installing additional modules ...
vboxadd.sh: Starting the VirtualBox Guest Additions.
Press Return to close this window...
```

A screenshot of a terminal window titled 'Терминал'. The window displays the command-line output of a script named 'vboxadd.sh' which is part of the VirtualBox Guest Additions installer. The output shows the verification of archive integrity, uncompression of the guest additions, and the copying of additional installer modules. It then installs these modules and starts the VirtualBox Guest Additions. Finally, it prompts the user to press 'Return' to close the window.

Рисунок 38 – процесс установки дополнений

После всех процедур требуется перезагрузить систему (рисунок 39). Чтобы перезагрузить систему, щелкните значок питания в правом верхнем углу.

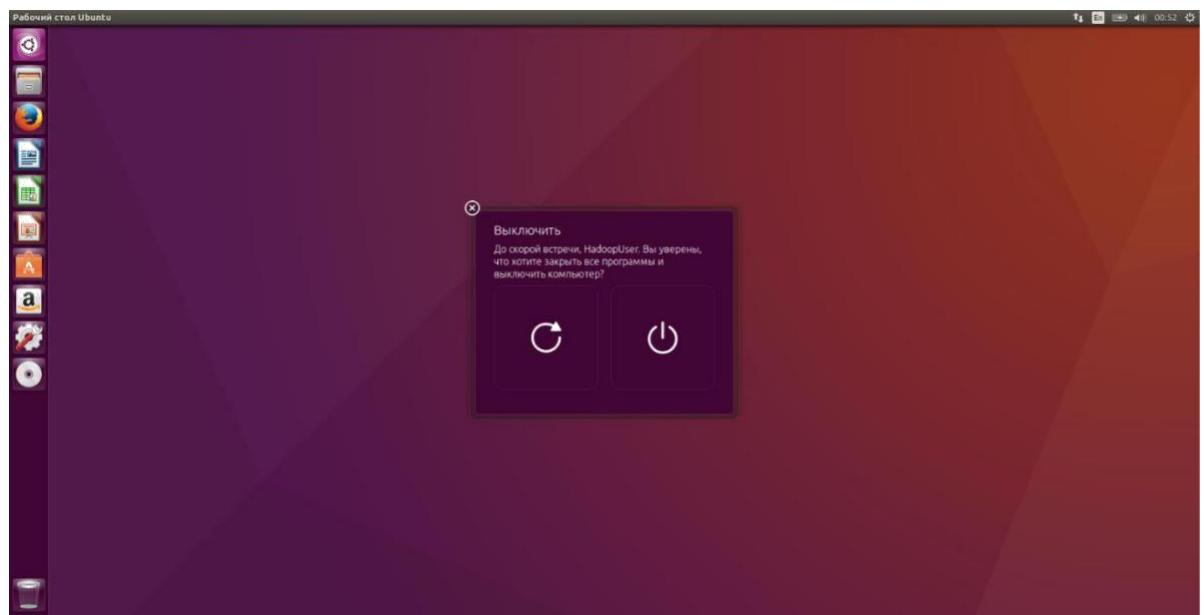


Рисунок 39 – окно перезагрузки операционной системы

Установка операционной системы завершена. Можно приступать к развертыванию платформы «Hadoop».

2.1.3 УСТАНОВКА ЭКОСИСТЕМЫ «HADOOP»

Теперь мы можем непосредственно приступать к установки экосистемы «Hadoop».

Как говорилось в начале практической части, скачивание дистрибутива «Hadoop» следует начинать на установленной системе «Linux» (так как передача файлов из «Windows 10» на виртуальную машину довольно затруднительный процесс).

После того как вы скачаете дистрибутив «Hadoop», можно начинать установку.

Основные пункты, которые будут рассматриваться в данной главе:

- Установка необходимого ПО (программного обеспечения).
- Установка и настройка компонентов «Hadoop».
- Запуск «Hadoop».

2.1.3.1 УСТАНОВКА НЕОБХОДИМОГО ПО

Для корректной работы экосистемы «Hadoop» требуется установить дополнительное программное обеспечение.

Откройте терминал системы «Linux» (консоль) и введите команды, указанные на рисунке 40.

```

hadoopuser@HadoopVM:~$ sudo apt-get install python-software-properties
[sudo] пароль для hadoopuser:
Чтение списков пакетов... Готово
Построение дерева зависимостей
Чтение информации о состоянии... Готово
Будут установлены следующие дополнительные пакеты:
  python-apt python-pycurl
Предлагаемые пакеты:
  python-apt-dbg python-apt-doc libcurl4-gnutls-dev python-pycurl-dbg python-pycurl-doc
Новые пакеты, которые будут установлены:
  python-apt python-pycurl python-software-properties
обновлено 0, установлено 3 новых пакетов, для удаления отмечено 0 пакетов, и 185 пакетов не обновлено.
Необходимо скачать 203 kB архивов.
После данной операции, объём занятого дискового пространства возрастёт на 930 kB.
Хотите продолжить? [Д/н] Д
Пол:1 http://ru.archive.ubuntu.com/ubuntu xenial/main amd64 python-apt amd64 1.1.0-beta1build1 [139 kB]
Пол:2 http://ru.archive.ubuntu.com/ubuntu xenial/main amd64 python-pycurl amd64 7.43.0-1ubuntu1 [43,3 kB]
Пол:3 http://ru.archive.ubuntu.com/ubuntu xenial-updates/universe amd64 python-software-properties all 0.96.20.7 [20,7 kB]
Получено 203 kB за 0с (1 023 kB/c).
Выбор ранее не выбранного пакета python-apt.
(Чтение базы данных ... на данный момент установлено 177355 файлов и каталогов.)
Подготовка к распаковке .../python-apt_1.1.0-beta1build1_amd64.deb ...
Распаковывается python-apt (1.1.0-beta1build1) ...
Выбор ранее не выбранного пакета python-pycurl.
Подготовка к распаковке .../python-pycurl_7.43.0-1ubuntu1_amd64.deb ...
Распаковывается python-pycurl (7.43.0-1ubuntu1) ...
Выбор ранее не выбранного пакета python-software-properties.
Подготовка к распаковке .../python-software-properties_0.96.20.7_all.deb ...
Распаковывается python-software-properties (0.96.20.7) ...
Настраивается пакет python-apt (1.1.0-beta1build1) ...
Настраивается пакет python-pycurl (7.43.0-1ubuntu1) ...
Настраивается пакет python-software-properties (0.96.20.7) ...
hadoopuser@HadoopVM:~$ sudo add-apt-repository ppa:webupd8team/java

```

Рисунок 40 – команды установки «Python» и хранилища «Java»

Описание команд с рисунка 40:

- Команда «`sudo apt-get install python-software-properties`» отвечает за установку программного обеспечения «Python».

— Команда «`sudo add-apt-repository ppa:webupdate/java`» отвечает за создания репозитория (хранилища) под программное обеспечение «Java».

Следующим шагом будет непосредственная установка программной платформы «Java». Для этого введем следующие команды, изображенные на рисунке 41.

```
hadoopuser@HadoopVM:~$ sudo apt-get update
Сущ:1 http://ru.archive.ubuntu.com/ubuntu xenial InRelease
Сущ:2 http://ru.archive.ubuntu.com/ubuntu xenial-updates InRelease
Сущ:3 http://ru.archive.ubuntu.com/ubuntu xenial-backports InRelease
Пол:4 http://ppa.launchpad.net/webupd8team/java/ubuntu xenial InRelease [17,6 kB]
Пол:5 http://ppa.launchpad.net/webupd8team/java/ubuntu xenial/main amd64 Packages [2 892 B]
Пол:6 http://ppa.launchpad.net/webupd8team/java/ubuntu xenial/main i386 Packages [2 892 B]
Пол:7 http://ppa.launchpad.net/webupd8team/java/ubuntu xenial/main Translation-en [1 260 B]
Пол:8 http://security.ubuntu.com/ubuntu xenial-security InRelease [102 kB]
Получено 127 kB за 3с (35,6 kB/c)
Чтение списков пакетов... Готово
hadoopuser@HadoopVM:~$ sudo apt-get install oracle-java8-installer
```

Рисунок 41 – установка платформы «Java»

После установки «Java», можно проверить ее версию, введя команду: «`java -version`» (Рисунок 42).

Теперь следует установить и настроить «SSH» — сетевой протокол прикладного уровня, позволяющий производить удалённое управление операционной системой и туннелирование TCP-соединений (например, для передачи файлов).

Для этого введем следующую команду, изображенную на рисунке 42.

```
hadoopuser@HadoopVM:~$ java -version
java version "1.8.0_131"
Java(TM) SE Runtime Environment (build 1.8.0_131-b11)
Java HotSpot(TM) 64-Bit Server VM (build 25.131-b11, mixed mode)
hadoopuser@HadoopVM:~$ sudo apt-get install openssh-server openssh-client
```

Рисунок 42 – установка «SSH»

Теперь следует настроить «SSH». Для этого введите команды, изображенные на рисунке 43.

```

hadoopuser@HadoopVM:~$ ssh-keygen -t rsa -P ""
Generating public/private rsa key pair.
Enter file in which to save the key (/home/hadoopuser/.ssh/id_rsa):
Created directory '/home/hadoopuser/.ssh'.
Your identification has been saved in /home/hadoopuser/.ssh/id_rsa.
Your public key has been saved in /home/hadoopuser/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:qtc1ge36RxPpFnS/9XGbghoenLA9LYVwtnQ4T/Qrdhg hadoopuser@HadoopVM
The key's randomart image is:
+---[RSA 2048]----+
|          o.        |
| . * oo .       |
| =oBE + .      |
| ..ooo* ..+    |
| S.+*.= 0     |
| o 0=oB. +.   |
| ...o*+ ..   |
| ... oo .    |
| .. . . .   |
+---[SHA256]-----+
hadoopuser@HadoopVM:~$ ls .ssh/
id_rsa id_rsa.pub
hadoopuser@HadoopVM:~$ cat $HOME/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_keys
hadoopuser@HadoopVM:~$ ssh localhost

```

Рисунок 43 – настройка конфигурации «SSH»

Описание команд с рисунка 43:

- Команда «ssh-keygen -t rsa -P “”» отвечает за генерацию пар ключей.
- Команда «ls .ssh/» позволяет пользователю посмотреть список файлов в папке.
 - Команда «cat \$HOME/.ssh/id_rsa.pub >> \$HOME/.ssh/authorized_keys» отвечает за конфигурирование «SSH» на беспарольной подключение к локальному компьютеру.
- Команда «ssh localhost» позволяет проверить удаленное подключение к локальному компьютеру.

2.1.3.2 УСТАНОВКА И НАСТРОЙКА КОМПОНЕНТОВ «HADOOP»

При скачивании дистрибутива «Hadoop», следует сохранить его на рабочий стол. Нам требуется перенести его в нашу домашнюю папку «/home/hadoopuser» и распаковать. Для этого введите команды, изображенные на рисунке 44.

```
hadoopuser@HadoopVM:~$ pwd  
/home/hadoopuser  
hadoopuser@HadoopVM:~$ mv "Рабочий стол"/hadoop-2.6.0-cdh5.11.0.tar.gz .  
hadoopuser@HadoopVM:~$ ls  
examples.desktop      Документы    Музыка        Шаблоны  
hadoop-2.6.0-cdh5.11.0.tar.gz  Загрузки    Общедоступные  
Видео                  Изображения Рабочий стол  
hadoopuser@HadoopVM:~$ tar xzf hadoop-2.6.0-cdh5.11.0.tar.gz  
hadoopuser@HadoopVM:~$ nano ~/.bashrc
```

Рисунок 44 – перенос и распаковка архива «Hadoop»

На рисунке 44 также изображена команда для открытия файла «.bashrc».

Описание команд с рисунка 44:

- Команда «pwd» позволяет пользователю посмотреть путь к своей домашней папке.
- Команда «mv “Рабочий стол”/hadoop-2.6.0-cdh5.11.0.tar.gz .» отвечает за перенос архива с платформой «Hadoop» в нашу домашнюю папку.
- Команда «tar xzf Hadoop-2.6.0-cdh5.11.0.tar.gz» распаковывает архив в папку со всеми правами для пользователя.
- Команда «nano ~/.bashrc» отвечает за открытие документа «.bashrc».

После того, как мы открыли файл «.bashrc» соответствующей командой, нам требует сконфигурировать его, введя строчки, изображенные на рисунке 45.

The screenshot shows a terminal window titled "hadoopuser@HadoopVM: ~". The title bar also displays "GNU nano 2.5.3" and "Файл: /home/hadoopuser/.bashrc". The main area of the window contains the following code:

```
if ! shopt -oq posix; then
    if [ -f /usr/share/bash-completion/bash_completion ]; then
        . /usr/share/bash-completion/bash_completion
    elif [ -f /etc/bash_completion ]; then
        . /etc/bash_completion
    fi
fi

export HADOOP_PREFIX="/home/hadoopuser/hadoop-2.6.0-cdh5.11.0"
export PATH=$PATH:$HADOOP_PREFIX/bin
export PATH=$PATH:$HADOOP_PREFIX/sbin
export HADOOP_MAPRED_HOME=${HADOOP_PREFIX}
export HADOOP_COMMON_HOME=${HADOOP_PREFIX}
export HADOOP_HDFS_HOME=${HADOOP_PREFIX}
export YARN_HOME=${HADOOP_PREFIX}
```

At the bottom of the window, there is a menu bar with Russian labels: Помощь, Записать, Поиск, Вырезать, Выровнять, ТекПозиц, Выход, ЧитФайл, Замена, Отмен. выр, Словарь, К строке.

Рисунок 45 – открытый файл «.bashrc»

С помощью этих строк мы задаем пути к стандартным библиотекам «Hadoop». После этого требуется перезапустить терминал, чтобы наши изменения вступили в силу.

Теперь нам требуется сконфигурировать еще несколько важных файлов. Для открытия данных файлов введите команды, изображенные на рисунке 46.

The screenshot shows a terminal window with the following command history:

```
hadoopuser@HadoopVM:~$ cd hadoop-2.6.0-cdh5.11.0/
hadoopuser@HadoopVM:~/hadoop-2.6.0-cdh5.11.0$ cd etc/hadoop
hadoopuser@HadoopVM:~/hadoop-2.6.0-cdh5.11.0/etc/hadoop$ nano hadoop-env.sh
hadoopuser@HadoopVM:~/hadoop-2.6.0-cdh5.11.0/etc/hadoop$ nano core-site.xml
hadoopuser@HadoopVM:~/hadoop-2.6.0-cdh5.11.0/etc/hadoop$ nano hdfs-site.xml
hadoopuser@HadoopVM:~/hadoop-2.6.0-cdh5.11.0/etc/hadoop$ cp mapred-site.xml.template mapred-site.xml
hadoopuser@HadoopVM:~/hadoop-2.6.0-cdh5.11.0/etc/hadoop$ nano mapred-site.xml
hadoopuser@HadoopVM:~/hadoop-2.6.0-cdh5.11.0/etc/hadoop$ nano yarn-site.xml
hadoopuser@HadoopVM:~/hadoop-2.6.0-cdh5.11.0/etc/hadoop$ █
```

Рисунок 46 – команды для открытия необходимых файлов

После того, как мы открыли файл «hadoop-env.sh» необходимой командой, введите следующую строку конфигурации с рисунка 47.

```

# Licensed to the Apache Software Foundation (ASF) under one
# or more contributor license agreements. See the NOTICE file
# distributed with this work for additional information
# regarding copyright ownership. The ASF licenses this file
# to you under the Apache License, Version 2.0 (the
# "License"); you may not use this file except in compliance
# with the License. You may obtain a copy of the License at
#
#     http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.

# Set Hadoop-specific environment variables here.

# The only required environment variable is JAVA_HOME. All others are
# optional. When running a distributed configuration it is best to
# set JAVA_HOME in this file, so that it is correctly defined on
# remote nodes.

# The java implementation to use.
export JAVA_HOME=/usr/lib/jvm/java-8-oracle/

# The jsvc implementation to use. Jsvc is required to run secure datanodes
# that bind to privileged ports to provide authentication of data transfer
# protocol. Jsvc is not required if SASL is configured for authentication of
# data transfer protocol using non-privileged ports.
#export JSVC_HOME=${JSVC_HOME}

```

Рисунок 47 – конфигурирование файла «hadoop-env.sh» Данная строка конфигурации задает путь к библиотеке платформы «Java».

После открытия файла «core-site.xml», введите строки конфигурации с рисунка 48.

```

<configuration>
<property>
<name>fs.defaultFS</name>
<value>hdfs://localhost:9000</value>
</property>

<property>
<name>hadoop.tmp.dir</name>
<value>/home/hadoopuser/hdata</value>
</property>
</configuration>

```

Рисунок 48 – конфигурирование файла «core-site.xml»

Данные строки конфигурации задают репозиторий для хранения данных «HDFS».

После открытия файла «hdfs-site.xml», введите следующие строки конфигурации с рисунка 50.

```
<configuration>
    <property>
        <name>dfs.replication</name>
        <value>1</value>
    </property>
</configuration>
```

Рисунок 50 – конфигурирование файла «hdfs-site.xml»

Данные строки конфигурации задают коэффициент репликации (копирования) равным единицы, так как у нас всего один жесткий диск в системе.

После открытия файла «mapred-site.xml», введите следующие строки конфигурации с рисунка 51.

```
<configuration>
    <property>
        <name>mapreduce.framework.name</name>
        <value>yarn</value>
    </property>
</configuration>
```

Рисунок 51 – конфигурирование файла «mapred-site.xml»

Данные строки конфигурации задают фреймворк «YARN» для реализации «MapReduce» задач.

После открытия файла «yarn-site.xml», введите следующие строки конфигурации с рисунка 52.

```
<configuration>
  <!-- Site specific YARN configuration properties -->
  <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>
</configuration>
```

Рисунок 52 – конфигурирование файла «yarn-site.xml»

Данные строки конфигурации позволяют нам подключить нужные библиотеки к фреймворку «YARN» для корректной работы.

Мы закончили развертывание и настройку компонентов экосистемы «Hadoop».

2.1.3.3 ЗАПУСК «HADOOP»

Теперь запустим «Hadoop» на нашей системе.

Для этого нам требуется перейти в корень папки «Hadoop» и форматировать «NameNode» файловой системы с помощью команд. Для этого используйте команды, изображенные на рисунке 53.

```
hadoopuser@HadoopVM:~$ cd hadoop-2.6.0-cdh5.11.0/
hadoopuser@HadoopVM:~/hadoop-2.6.0-cdh5.11.0$ bin/hdfs namenode -format
```

Рисунок 53 – команды форматирования

Следует знать, что команду форматирования следует вводить только один раз (после установки «Hadoop»), иначе мы можем потерять все данные из «HDFS».

После того, как прошло форматирование «NameNode», нужно удостовериться в успешном завершении данного действия.

Для этого необходимо найти строчку, изображенную на рисунке 54.

```
17/06/06 03:27:05 INFO common.Storage: Storage directory /home/hadoopuser/hdata/dfs/name has been successfully formatted.
```

Рисунок 54 – строка успешного завершения форматирования

Теперь можно запускать компоненты экосистемы «Hadoop». Для этого введите команды. Изображенные на рисунке 55.

```
hadoopuser@HadoopVM:~/hadoop-2.6.0-cdh5.11.0$ pwd
/home/hadoopuser/hadoop-2.6.0-cdh5.11.0
hadoopuser@HadoopVM:~/hadoop-2.6.0-cdh5.11.0$ sbin/start-dfs.sh
17/06/06 03:32:01 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Starting namenodes on [localhost]
localhost: starting namenode, logging to /home/hadoopuser/hadoop-2.6.0-cdh5.11.0/logs/hadoop-hadoopuser-namenode-HadoopVM.out
localhost: starting datanode, logging to /home/hadoopuser/hadoop-2.6.0-cdh5.11.0/logs/hadoop-hadoopuser-datanode-HadoopVM.out
Starting secondary namenodes [0.0.0.0]
The authenticity of host '0.0.0.0 (0.0.0.0)' can't be established.
ECDSA key fingerprint is SHA256:Pfgb5dYbx1NVFzxLTHeAYNzBzf+HdkCoUWqkn3hzsVo.
Are you sure you want to continue connecting (yes/no)? yes
0.0.0.0: Warning: Permanently added '0.0.0.0' (ECDSA) to the list of known hosts.
0.0.0.0: starting secondarynamenode, logging to /home/hadoopuser/hadoop-2.6.0-cdh5.11.0/logs/hadoop-hadoopuser-secondarynamenode-HadoopVM.out
17/06/06 03:32:52 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
hadoopuser@HadoopVM:~/hadoop-2.6.0-cdh5.11.0$ jps
3580 NameNode
3892 SecondaryNameNode
3692 DataNode
4013 Jps
hadoopuser@HadoopVM:~/hadoop-2.6.0-cdh5.11.0$ sbin/start-yarn.sh
starting yarn daemons
starting resourcemanager, logging to /home/hadoopuser/hadoop-2.6.0-cdh5.11.0/logs/yarn-hadoopuser-resourcemanager-HadoopVM.out
localhost: starting nodemanager, logging to /home/hadoopuser/hadoop-2.6.0-cdh5.11.0/logs/yarn-hadoopuser-nodemanager-HadoopVM.out
hadoopuser@HadoopVM:~/hadoop-2.6.0-cdh5.11.0$ jps
3580 NameNode
3892 SecondaryNameNode
4166 NodeManager
4859 ResourceManager
3692 DataNode
4461 Jps
hadoopuser@HadoopVM:~/hadoop-2.6.0-cdh5.11.0$ █
```

Рисунок 55 – команды запуска компонентов «Hadoop»

Чтобы удостовериться, что все модули запущены, введите команду «jps» с рисунка 55. В списке запущенных модулей должны присутствовать следующие наименования:

- «NameNode».
- «DataNode».
- «ResourceManager».
- «NodeManager».
- «Jps».
- «SecondaryNameNode».

Это означает, что экосистема «Hadoop» запущена и готова к работе.

Вы можете следить за работой платформы «Hadoop» через веб-браузер, введя в строку адреса: «localhost:50070» (Рисунок 56).

В этой же вкладке управления платформой можно посмотреть файлы в нашей файловой системе «HDFS».

The screenshot shows a web browser window titled 'Namenode information' with the URL 'localhost:50070/dfshealth.html&tab=overview'. The page has a dark blue header with tabs: 'Hadoop', 'Overview', 'Datanodes', 'Datanode Volume Failures', 'Snapshot', 'Startup Progress', and 'Utilities'. Below the header, there are two main sections: 'Overview' and 'Summary'. The 'Overview' section displays cluster statistics:

Started:	2.6.0-cdh5.11.0_r91a488f2c5abb3de0ebee74080dbc439c7576fb4
Version:	2.6.0-cdh5.11.0_r91a488f2c5abb3de0ebee74080dbc439c7576fb4
Compiled:	Thu Apr 06 08:07:00 +0500 2017 by jenkins from Unknown
Cluster ID:	CID-10302c6c-a1c2-41eb-be75-9a265527588a
Block Pool ID:	BP-1981509127-127.0.1.1-1496701625591

The 'Summary' section displays memory usage statistics:

Configured Capacity:	32.36 GB
DFS Used:	24 KB (0%)
Non DFS Used:	6.2 GB
DFS Remaining:	24.49 GB (75.69%)
Block Pool Used:	24 KB (0%)

At the bottom of the page, there is a note: 'Firefox automatically sends some data to Mozilla so that we can improve your experience.' and a 'Choose What I Share' button.

Рисунок 56 – веб-приложение для управления платформой «Hadoop»

На этом мы закончили установку и настройку экосистемы «Hadoop» на нашей виртуальной машине под управлением операционной системы «Linux».

Следующим шагом будет тестирование нашей системы с помощью программы подсчета слов в тексте «WordCount».

2.2 НАПИСАНИЕ ПРОГРАММЫ «WORDCOUNT» ДЛЯ ТЕСТИРОВАНИЯ СИСТЕМЫ «HADOOP»

Чтобы протестировать возможности экосистемы «Hadoop» мы будем использовать стандартный метод – программу для подсчета слов в тексте «WordCount».

Данная программа будет написана на языке программирования «Java» в среде разработки «Eclipse», которую требуется скачать с официального сайта.

Данная глава будет разделена на два этапа:

- Установка среды разработки «Eclipse».
- Написание кода программы «WordCount».
- Запуск программы «WordCount».

2.2.1 УСТАНОВКА СРЕДЫ РАЗРАБОТКИ «ECLIPSE»

После того, как вы скачали архив «Eclipse», распакуйте его.

Нажмите правой кнопкой мыши на архив и в выпадающем меню выберите пункт «Извлечь сюда» (Рисунок 57).

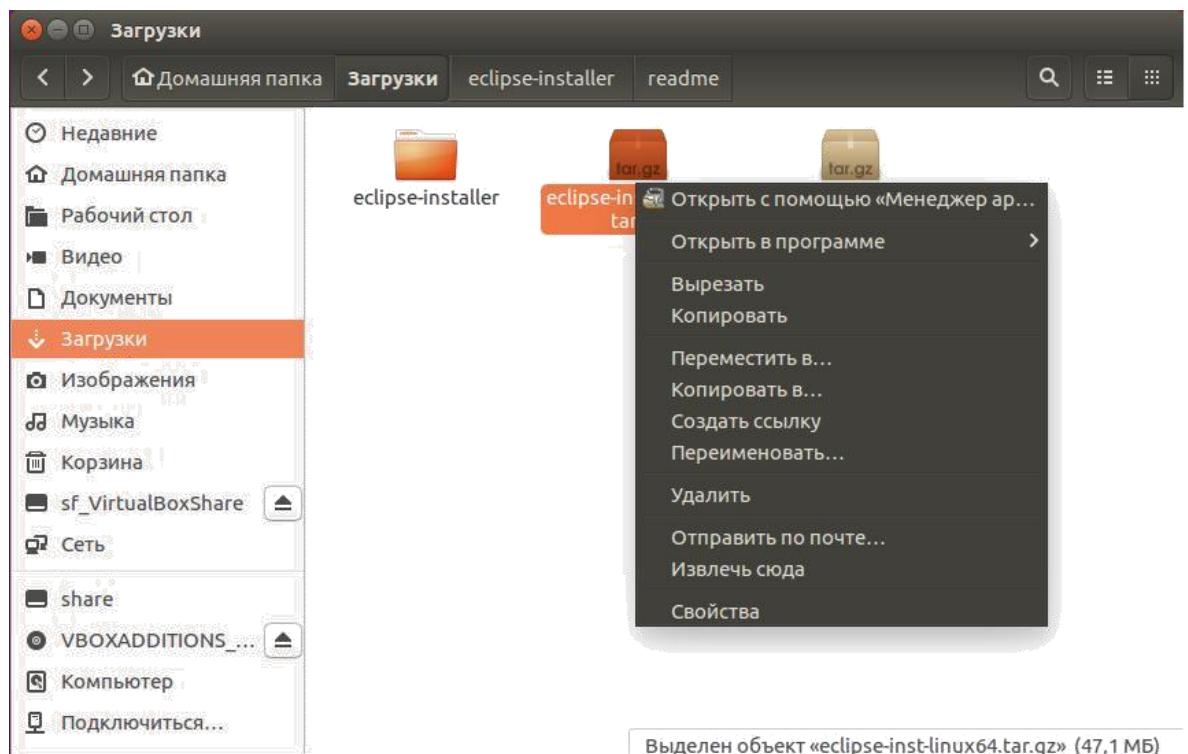


Рисунок 57 – извлечение архива «Eclipse»

Откройте терминал и перейдите в папку установщика «Eclipse», введя команду с рисунка 58.

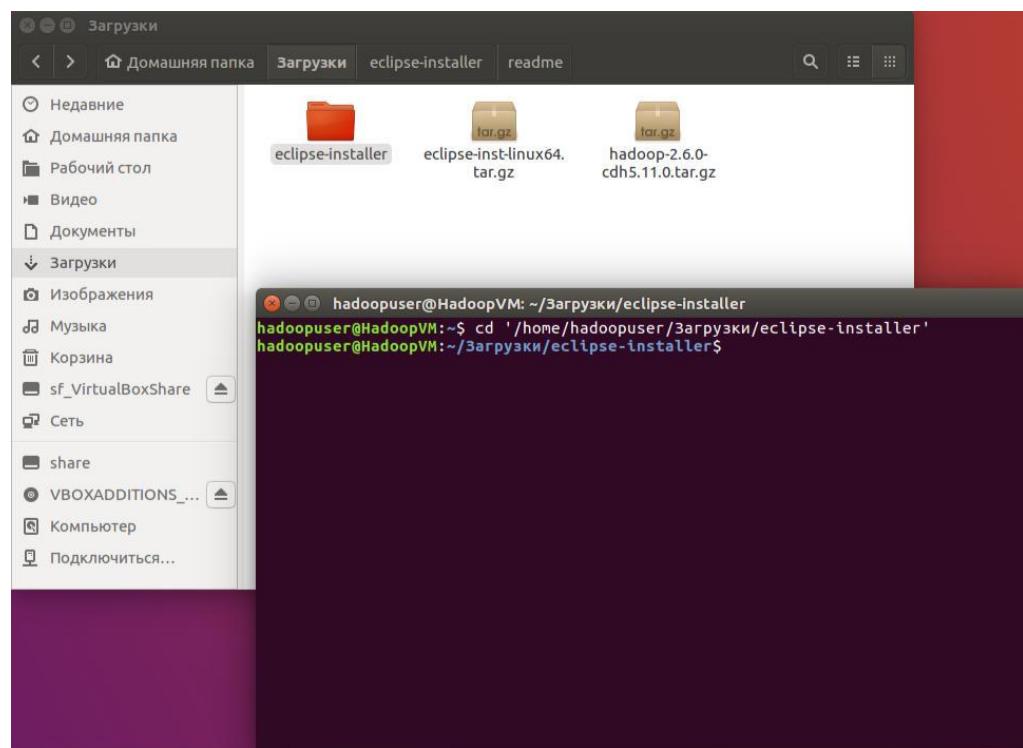


Рисунок 58 – переход в папку установщика «Eclipse» через терминал

Введите команду с рисунка 59, чтобы запустить файл установки среды разработки «Eclipse».

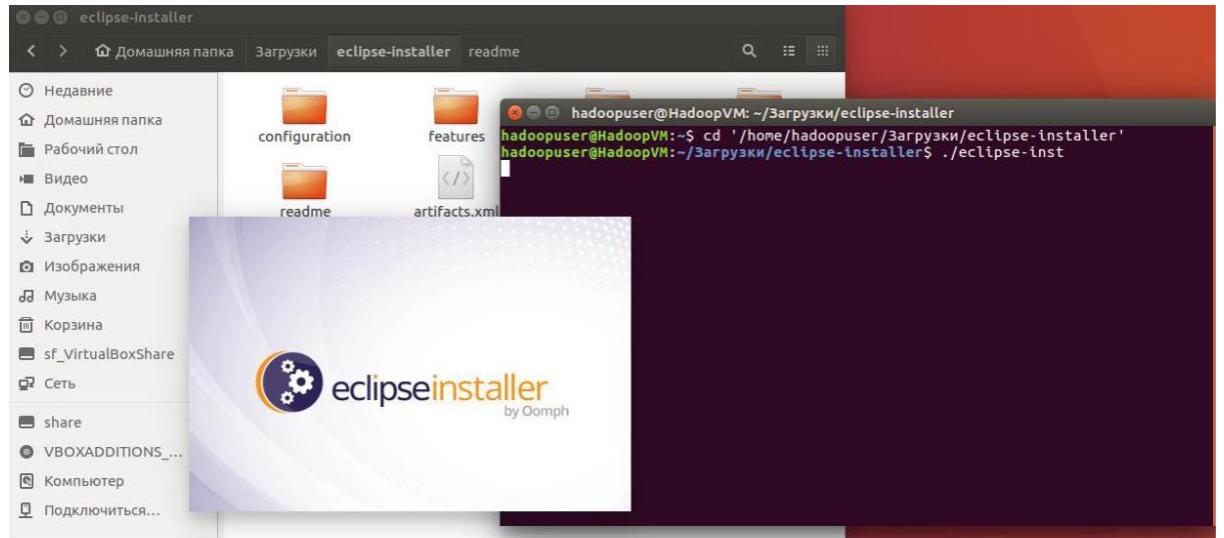


Рисунок 59 – запуск установщика «Eclipse»

В окне установщика «Eclipse» (Рисунок 60) выберите пункт «Eclipse IDE for Java Developers».

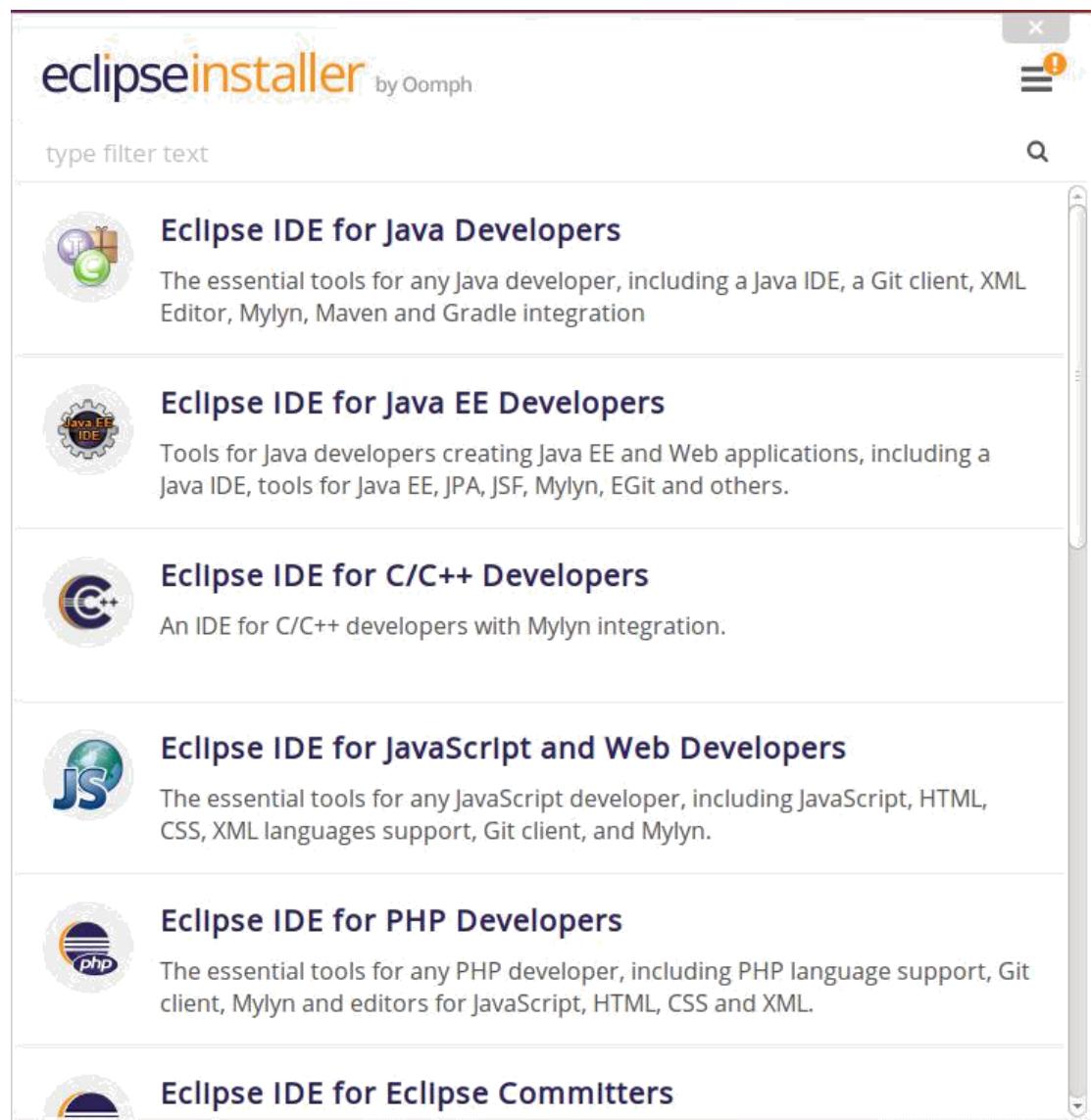


Рисунок 60 – окно установщика «Eclipse»

Оставьте стандартную папку для установки (Рисунок 61). Нажмите кнопку «Install».

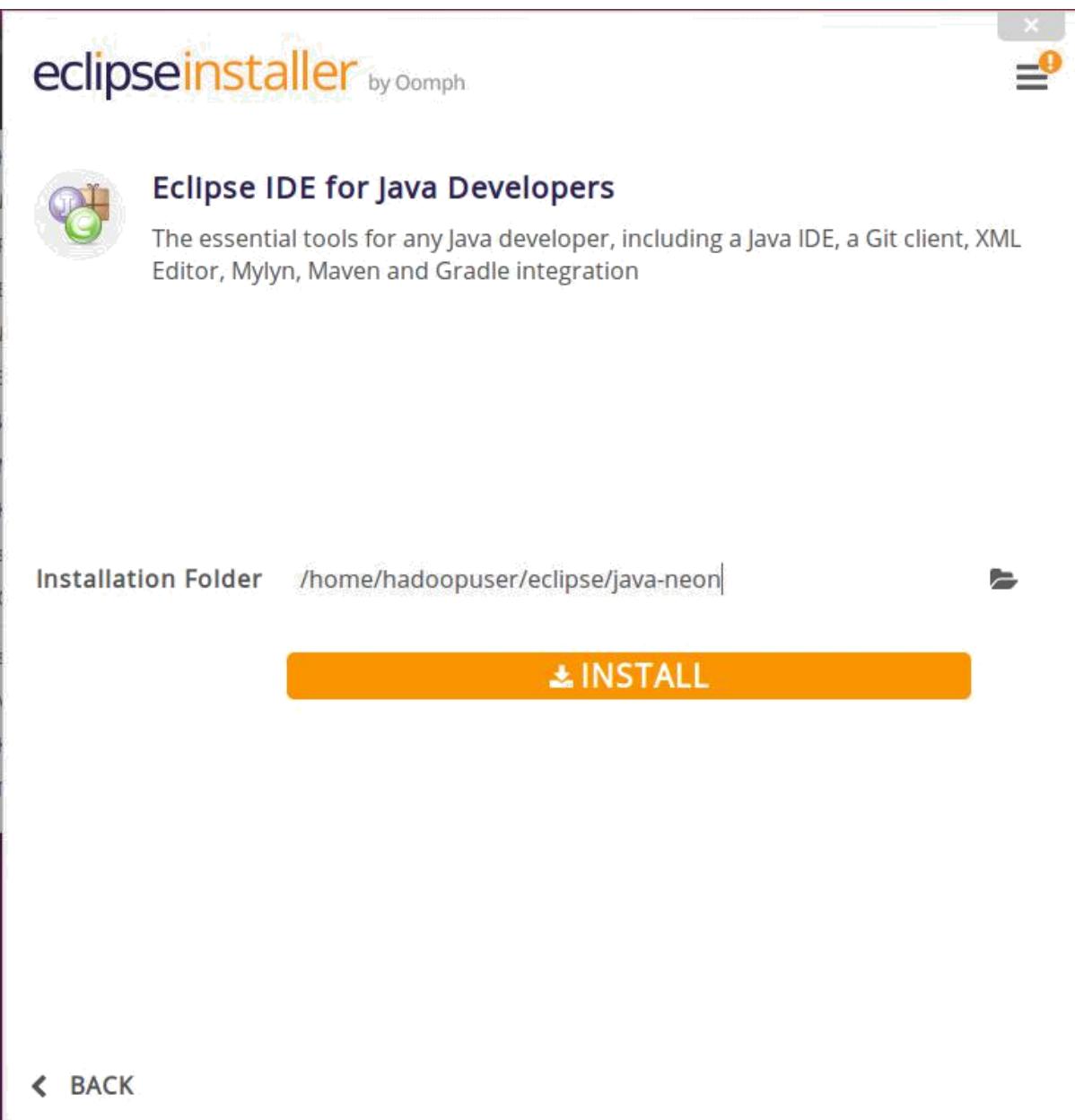


Рисунок 61 – настройка установки

«Eclipse» Дождитесь завершения установки (Рисунок 62).

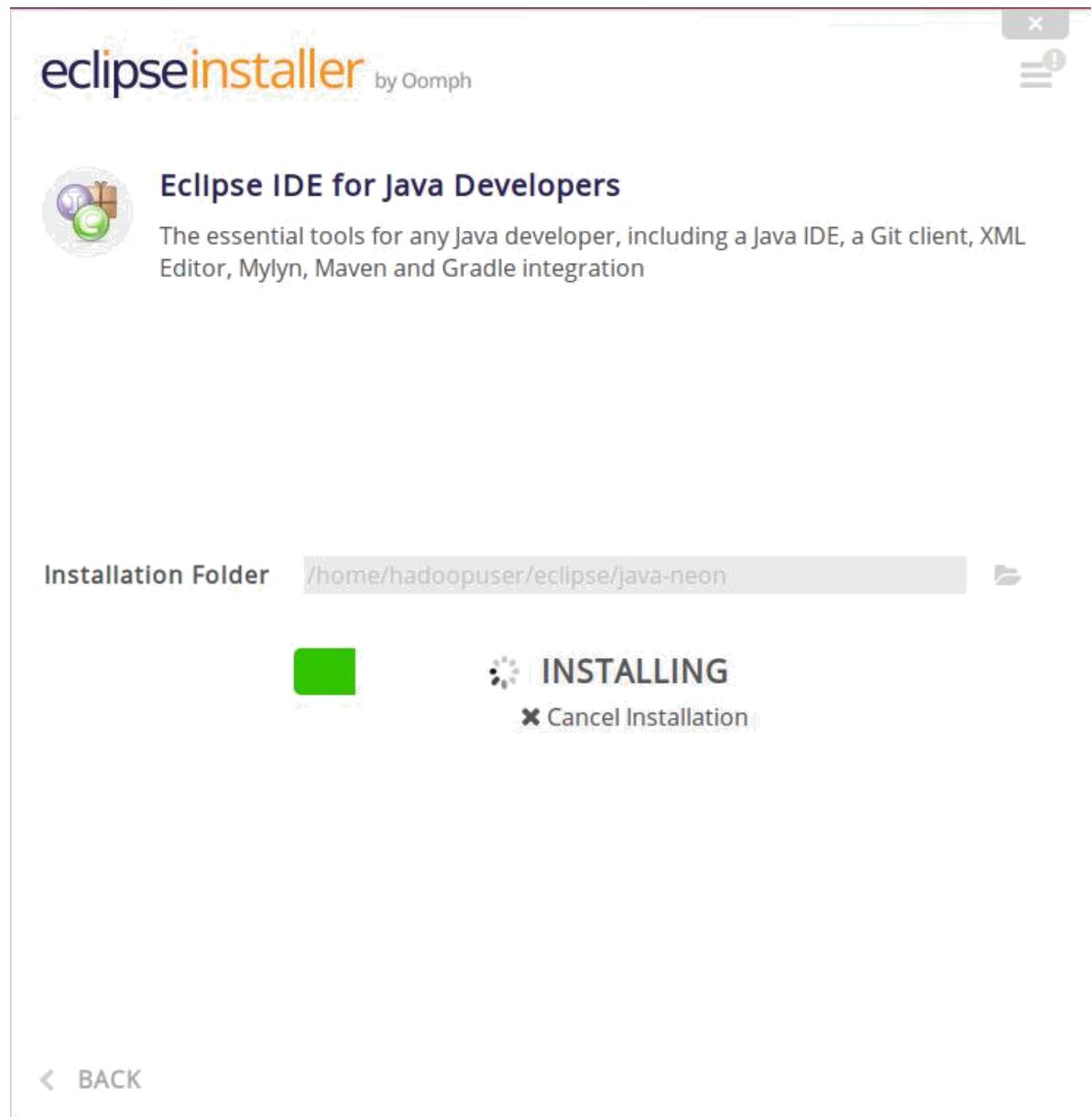


Рисунок 62 – процесс установки «Eclipse»

В появившемся окне лицензирования примите требования разработчика (Рисунок 63).

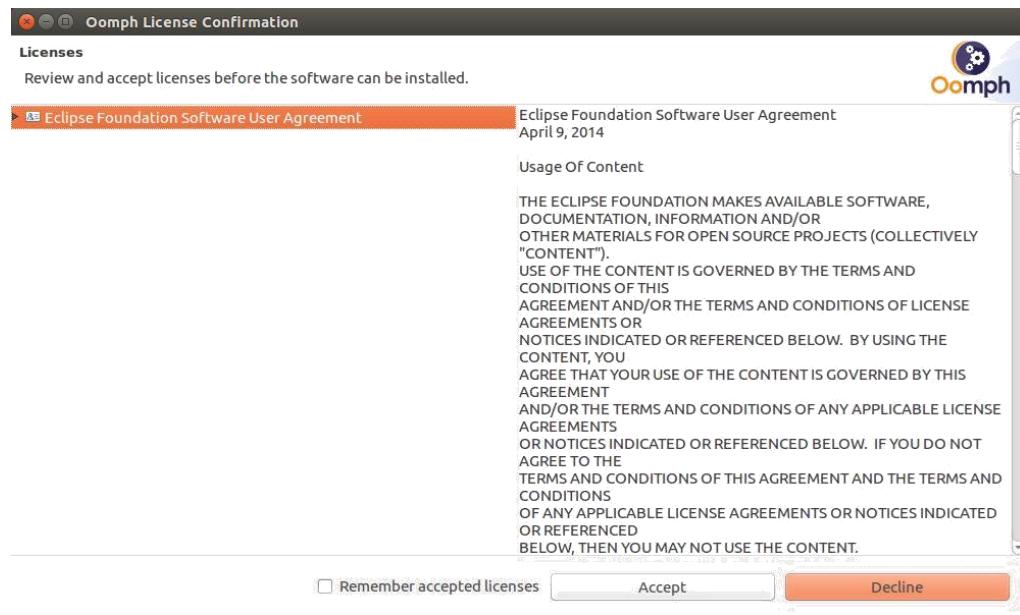


Рисунок 63 – окно лицензирования

Оставьте стандартный путь к рабочему пространству в окне настройки (Рисунок 64).

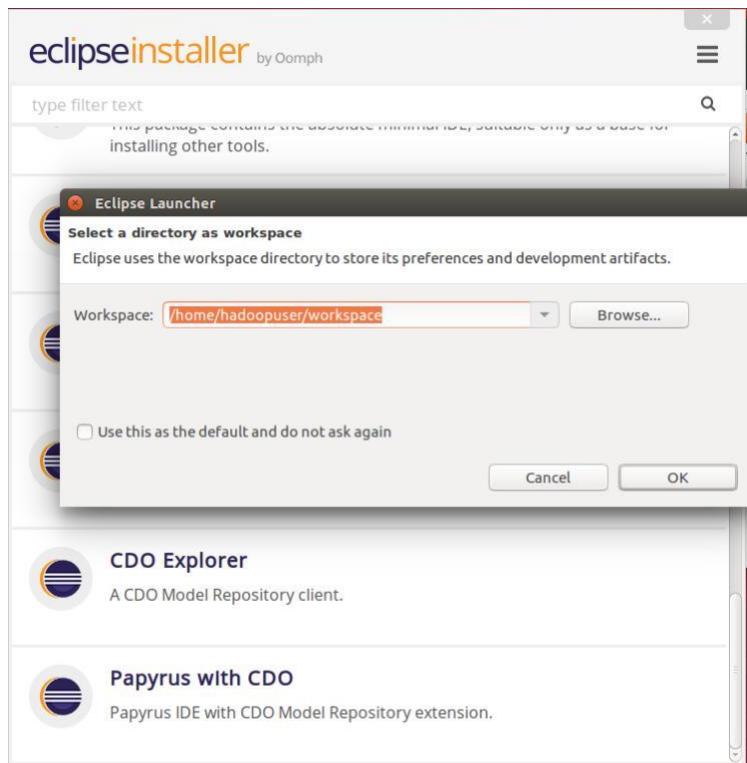


Рисунок 64 – настройка рабочего пространства «Eclipse»

Запустите программу, кликнув на иконку «eclipse» (Рисунок 65).



Рисунок 65 – запуск программы «Eclipse»

Откроется рабочее пространство (Рисунок 66).

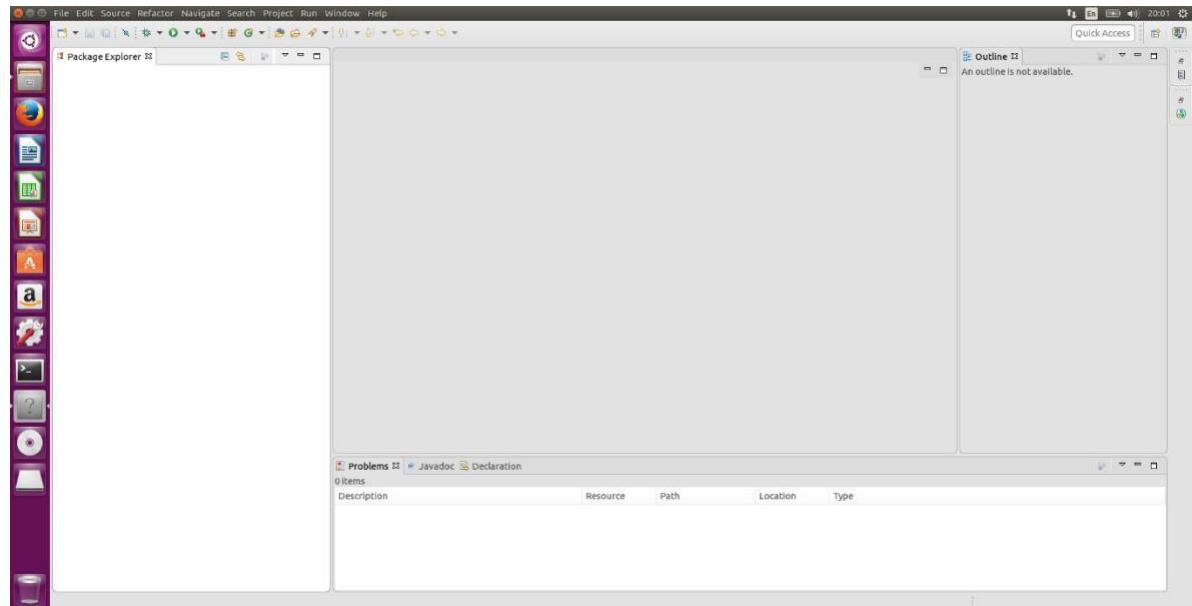


Рисунок 66 – рабочее пространство «Eclipse»

Мы завершили установку и запуск среды программирования «Eclipse».

2.2.2 НАПИСАНИЕ КОДА ПРОГРАММЫ «WORDCOUNT»

Для начала нужно создать проект.

Перейдите в рабочую среду «Eclipse».

Правой кнопкой мыши вызовите выпадающее меню и выберите пункт «New
-> Java Project» (Рисунок 67).

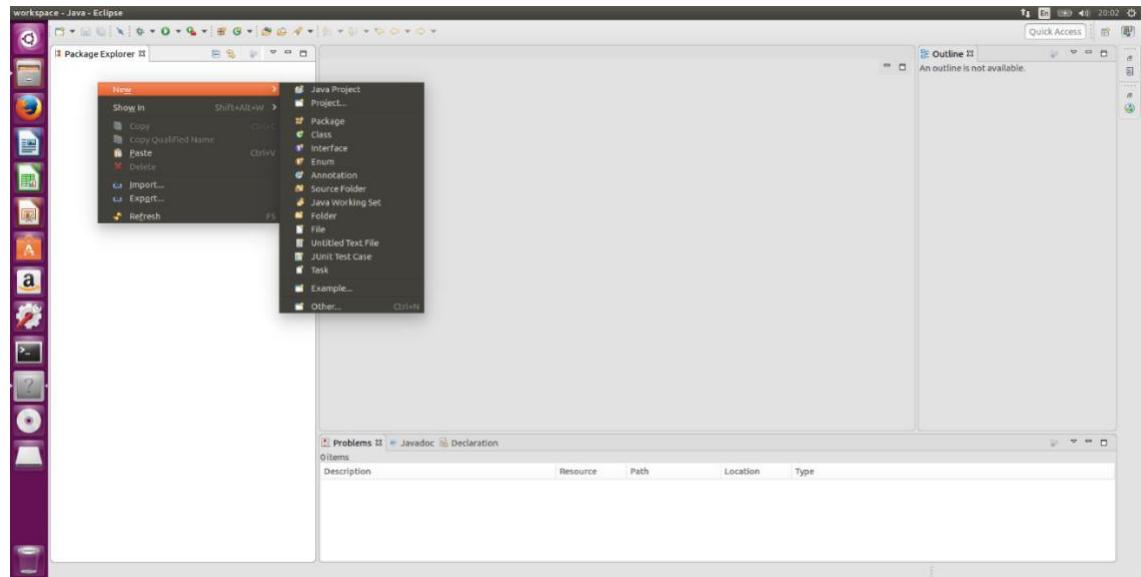


Рисунок 67 – создание проекта

Введите название проекта «WordCountJob» и нажмите кнопку «Finish».

После того, как вы создали проект, он отобразится в рабочей среде «Eclipse»
(Рисунок 68).

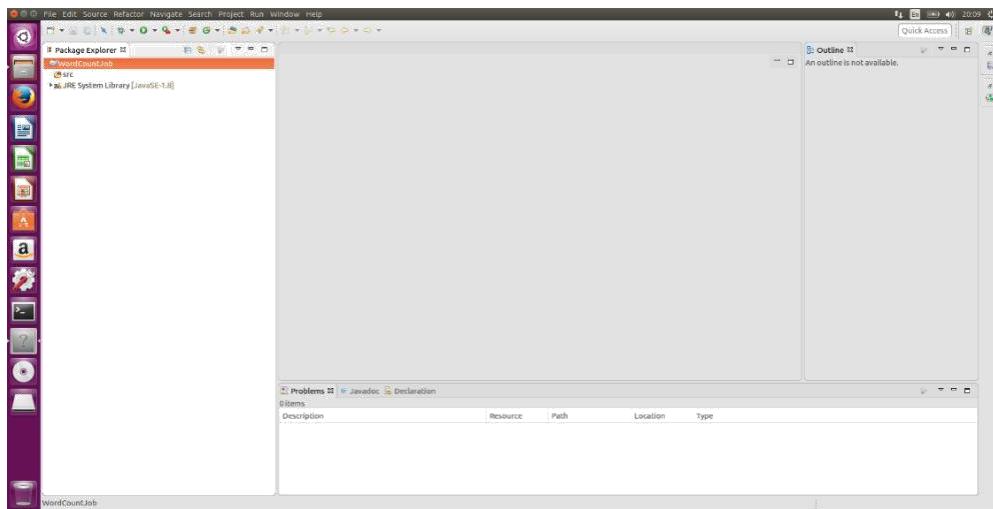


Рисунок 68 – созданный проект «WordCountJob»

Затем нам следует создать несколько классов:

- «WordCount».
- «WordCountMapper».
- «WordCountReducer».

Для этого перейдите в рабочую среду «Eclipse».

Нажмите правой кнопкой мыши на проект, вызвав выпадающее меню, и выберите пункт «New → Class» (Рисунок 69).

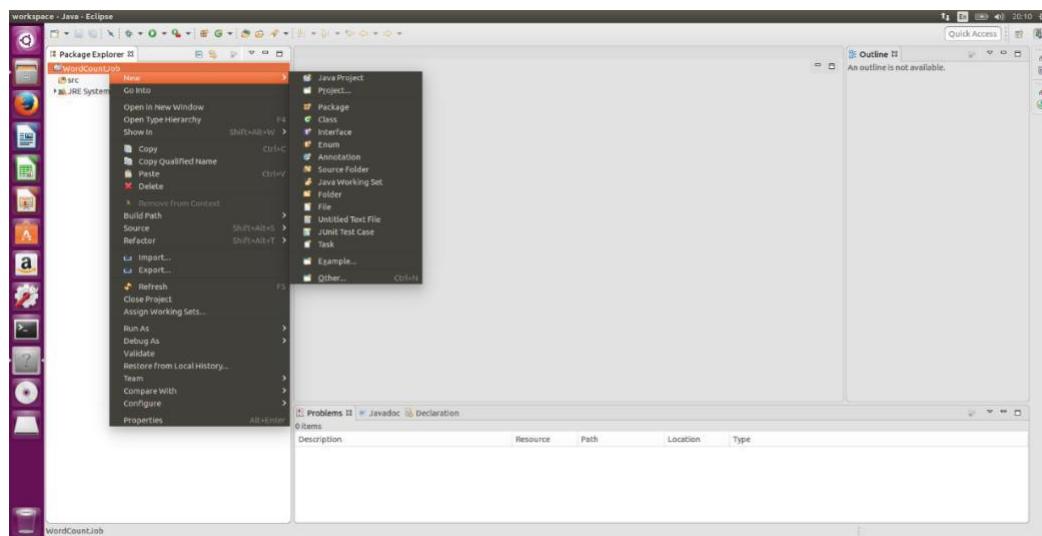


Рисунок 69 – создание классов

Пример создания класса «WordCount» показан на рисунке 70. После того, как вы указали имя класса, нажмите кнопку «Finish».

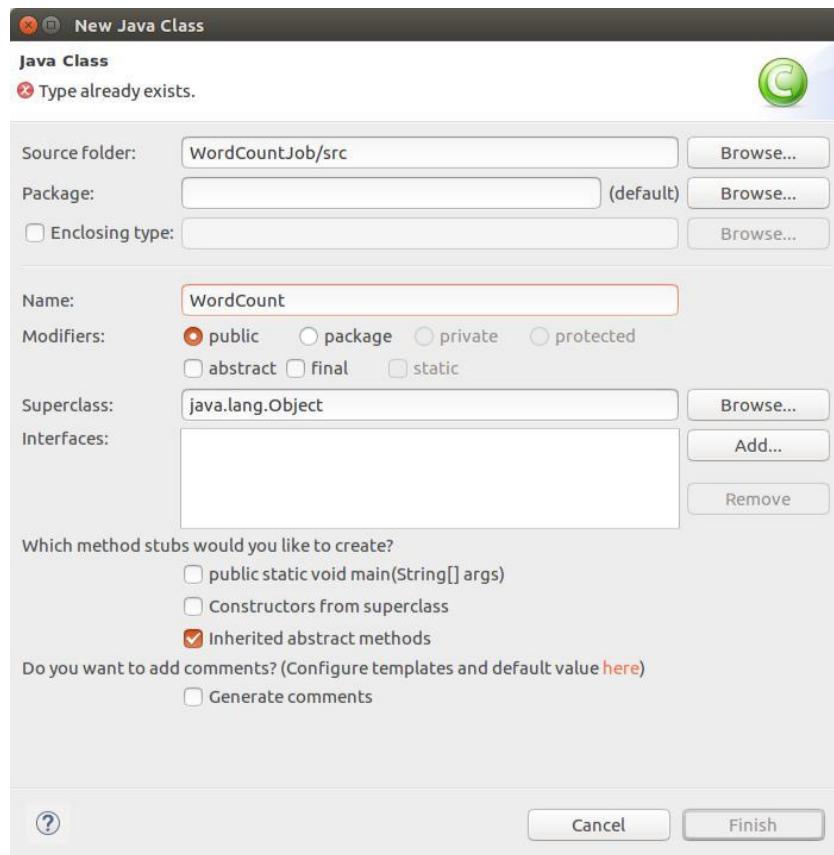


Рисунок 70 – создание класса «WordCount» программы

Теперь следует написать следующие строки кода в классах:

- В классе «WordCount» код из ПРИЛОЖЕНИЯ А.
- В классе «WordCountMapper» код из ПРИЛОЖЕНИЯ Б.
- В классе «WordCountReducer» код из ПРИЛОЖЕНИЯ В.

Далее требуется подключить библиотеки «Hadoop» к нашей программе для корректной работы. Для этого щелкните правой кнопкой мыши по проекту, выберите пункт «Properties» из выпадающего меню (Рисунок 71).

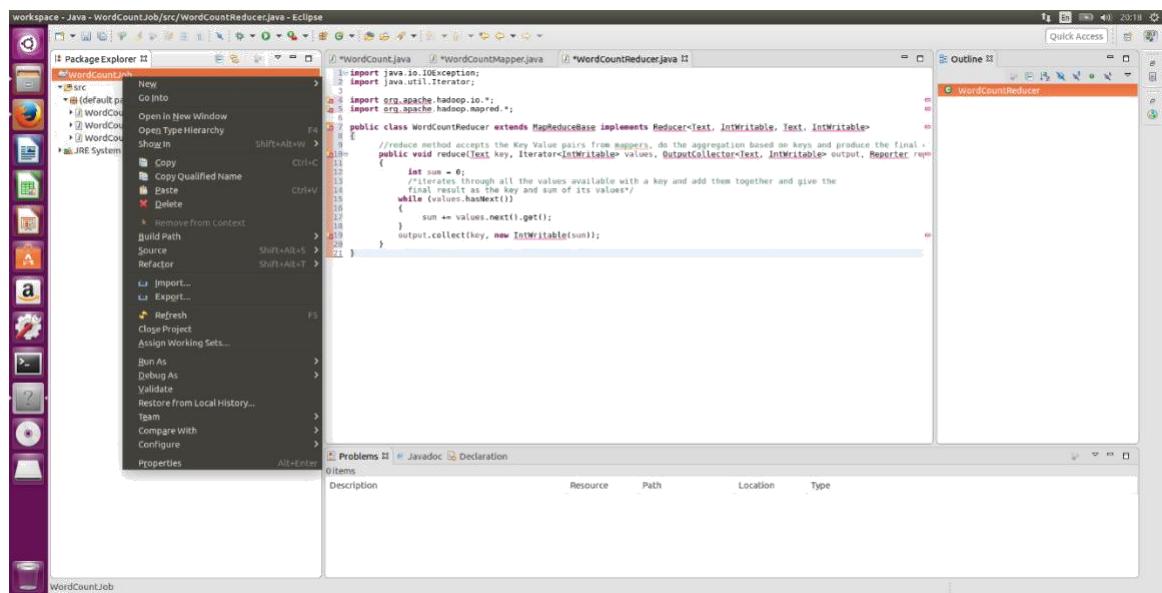


Рисунок 71 – выбор настроек проекта

В открывшемся окне настроек проекта выберите пункт «Java Build Path», перейдите на вкладку «Libraries» и нажмите кнопку «Add External JARs» (Рисунок 72).

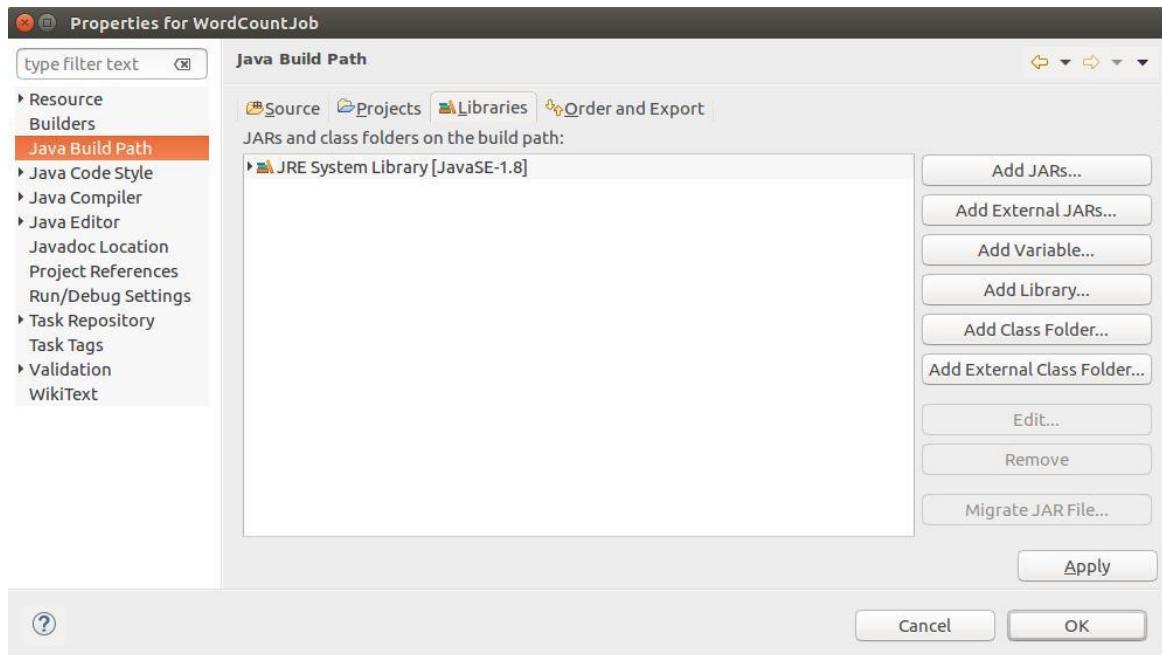


Рисунок 72 – окно подключения библиотек

Выберите все «jar» файлы , находящиеся в следующих местах (Рисунки 73, 74. 75, 76):

- «Hadoop–2.6.0–cdh5.11.0/share/hadoop/common».
- «Hadoop–2.6.0–cdh5.11.0/share/hadoop/common/lib».
- «Hadoop–2.6.0–cdh5.11.0/share/hadoop/mapreduce».
- «Hadoop–2.6.0–cdh5.11.0/share/hadoop/yarn».
- «Hadoop–2.6.0–cdh5.11.0/share/hadoop/hdfs».

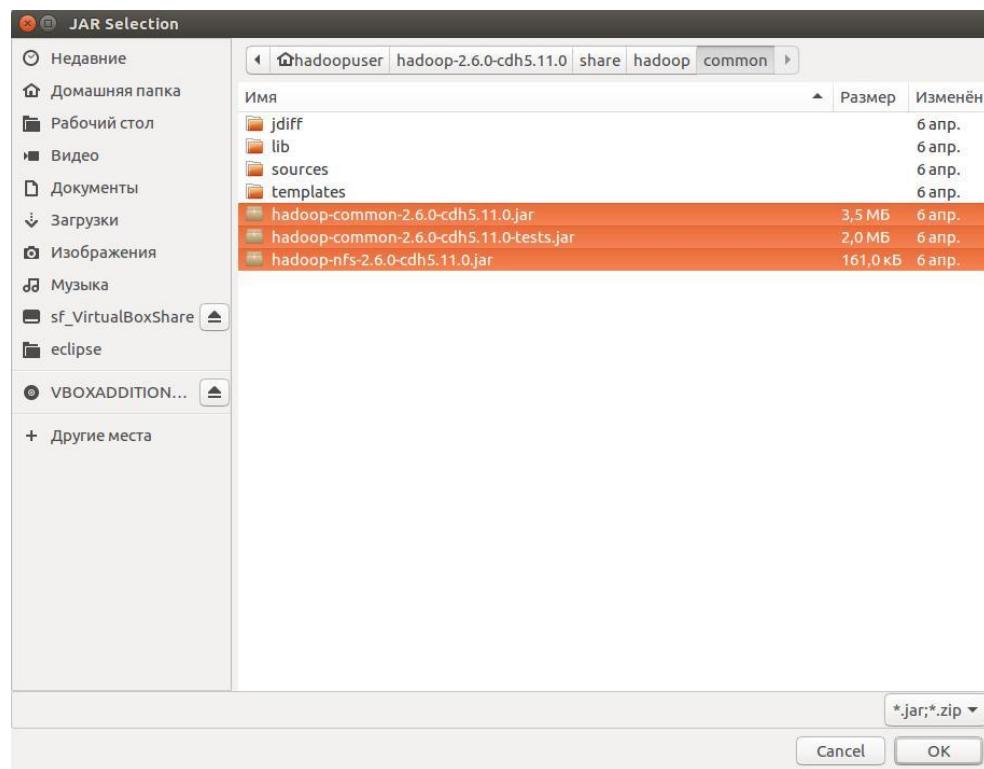


Рисунок 73 – добавление библиотек

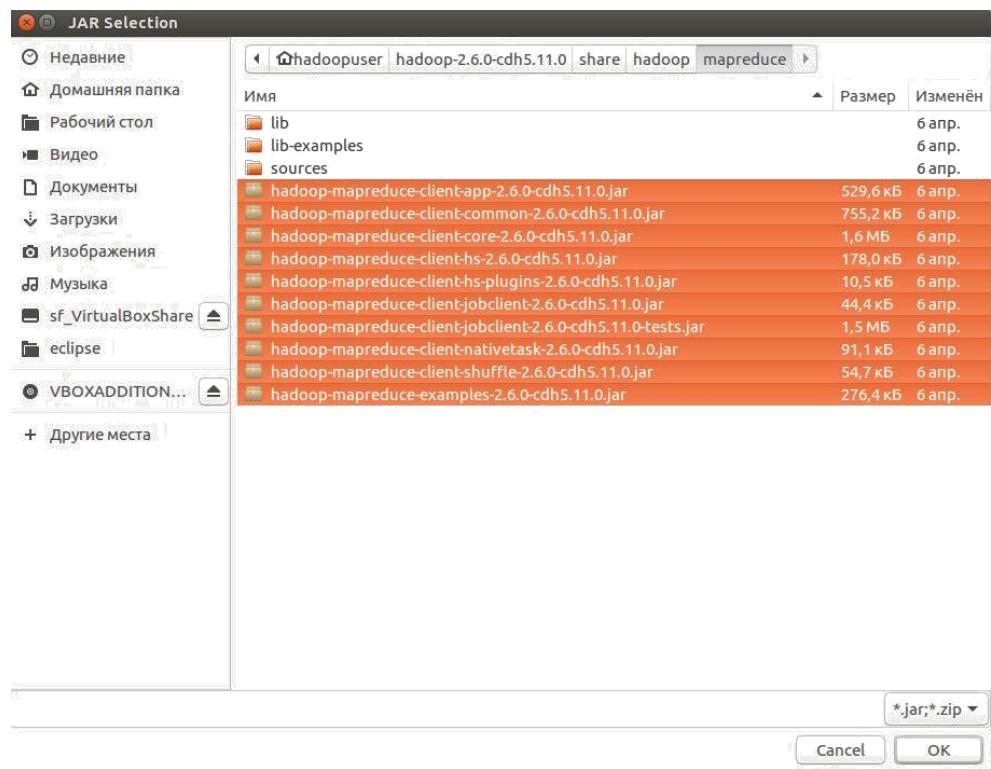


Рисунок 74 – добавление библиотек

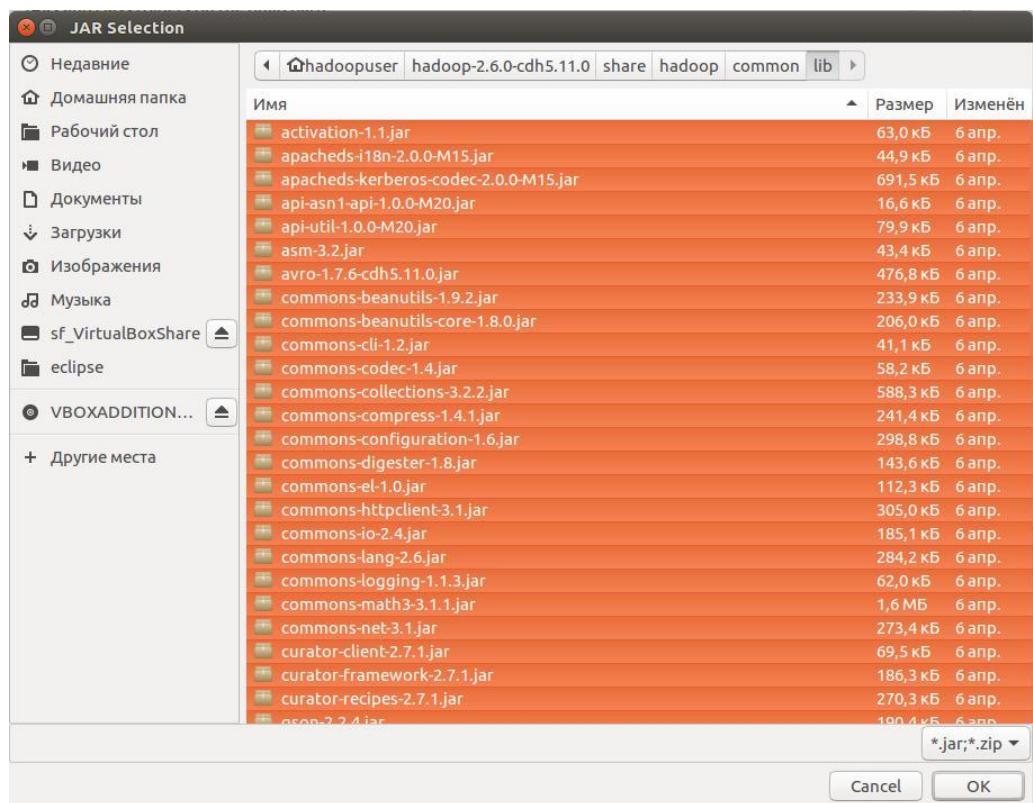


Рисунок 75 – добавление библиотек

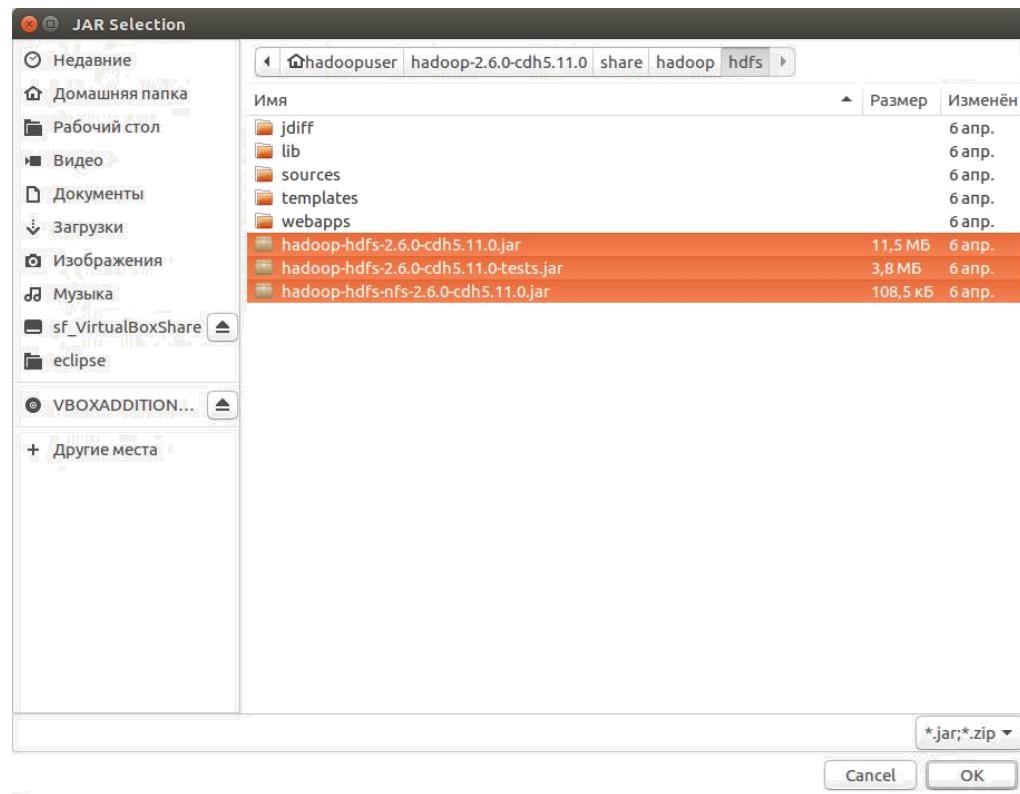


Рисунок 76 – добавление библиотек

После того, как вы добавили все нужные библиотеки нажмите кнопку «OK» (Рисунок 77).

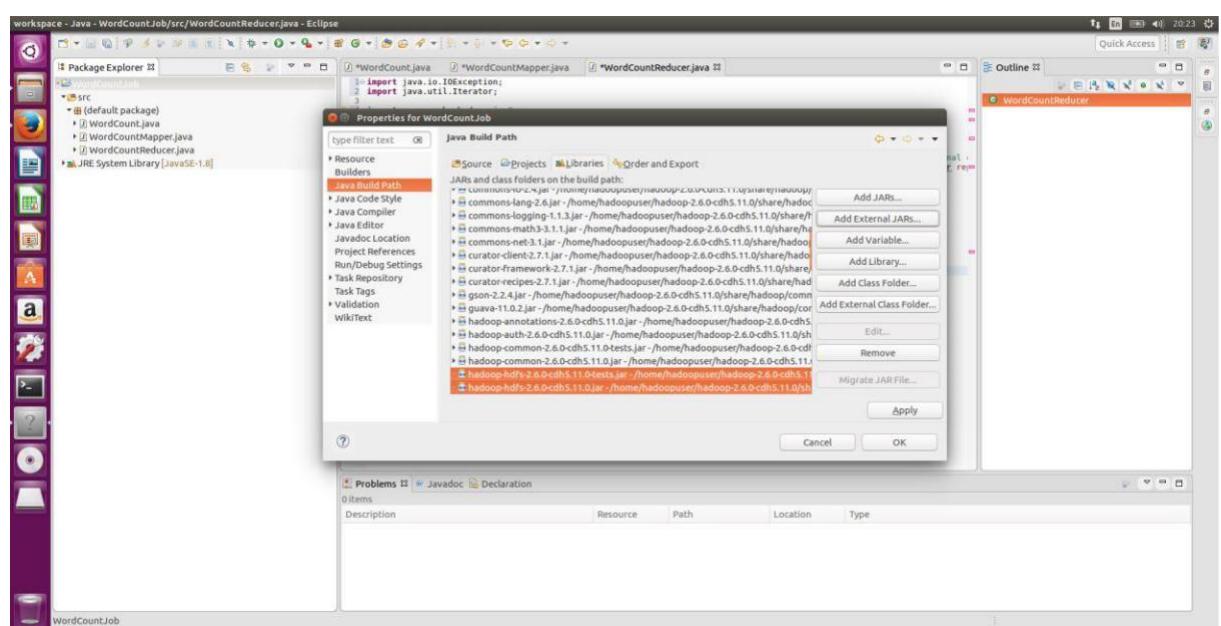


Рисунок 77 – окно добавления библиотек

Как только мы подключили все библиотеки, наш проект будет выглядеть следующим образом (Рисунок 78).

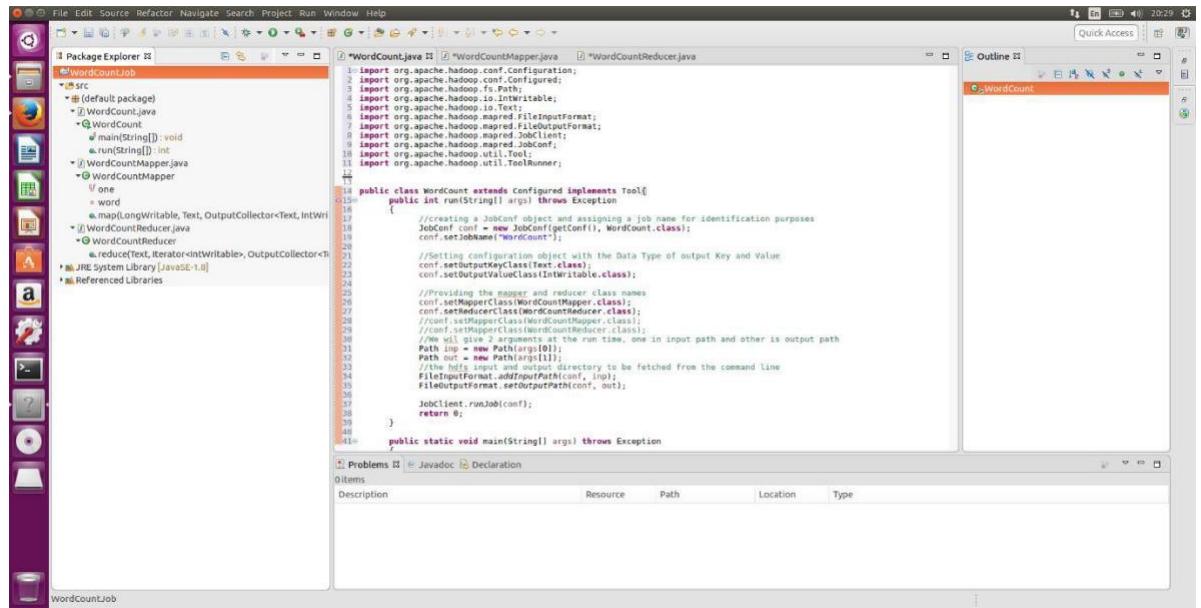


Рисунок 78 – вид проекта после добавления библиотек

Теперь требуется сконфигурировать нашу программу как «Java–приложение».

Для этого щелкните правой кнопкой мыши по проекту, выберите пункт «Properties» из выпадающего меню.

После этого, в открывшемся окне настроек выберите пункт «Run/Debug Settings». Выберите «Java Application» и нажмите кнопку «OK» (Рисунок 79).

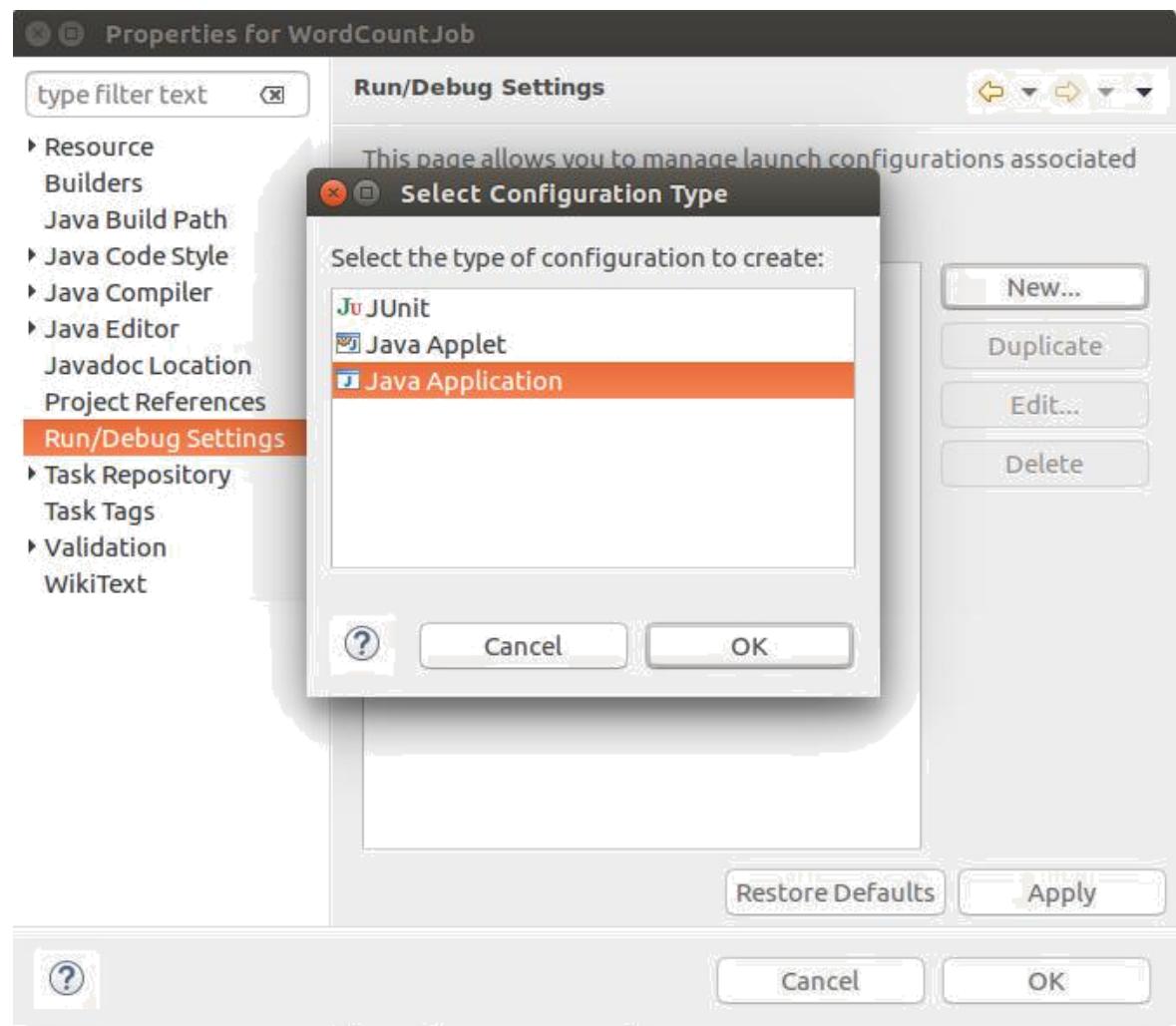


Рисунок 79 – окно создания «Java–приложения»

Далее требуется сконфигурировать приложения. В появившемся окне задайте имя «WordCountConfig» и главный класс «WordCount» (Рисунок 80). Нажмите кнопку «OK».

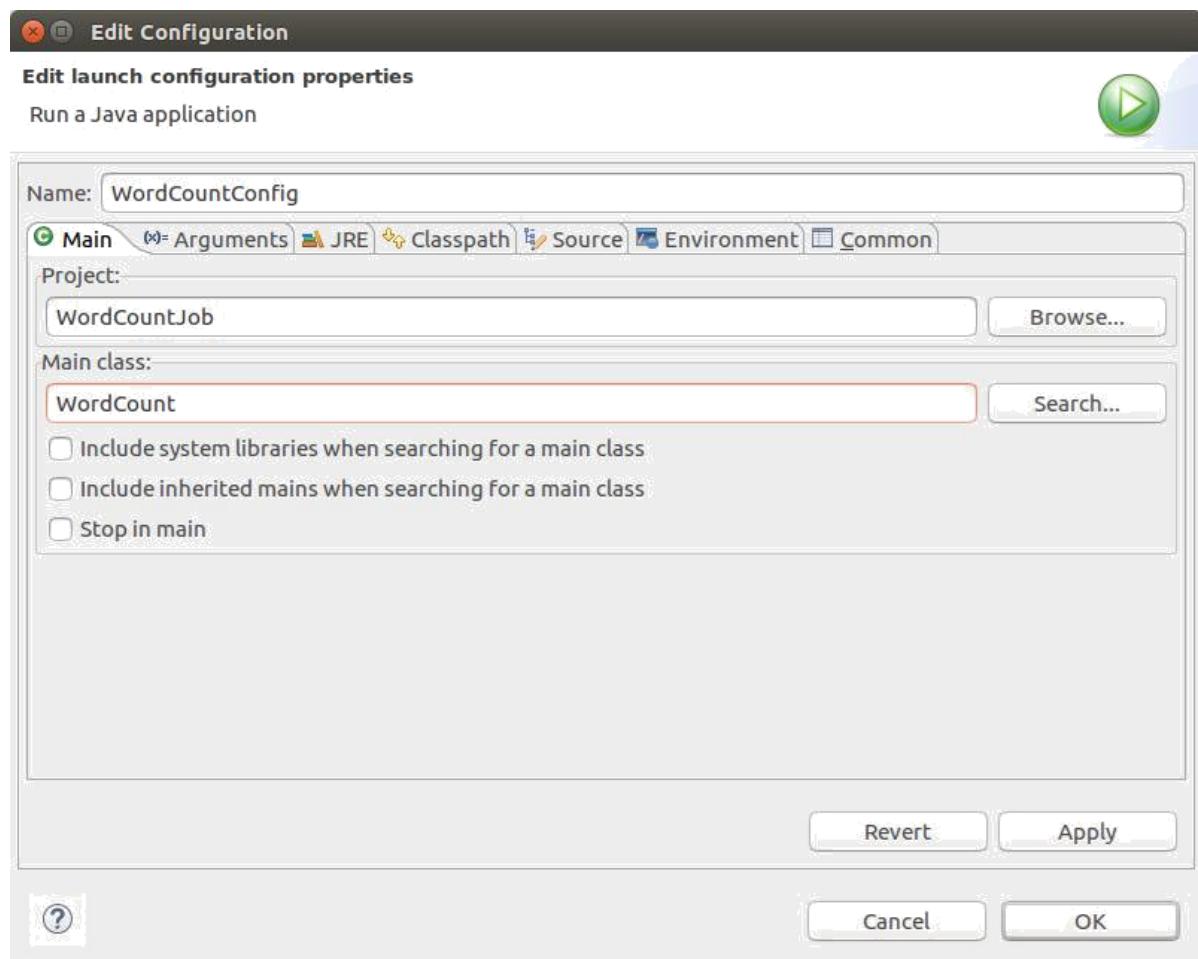


Рисунок 80 – конфигурация приложения

Далее нам требуется скомпоновать наше приложение в «jar» файл.

Для этого щелкните правой кнопкой мыши по проекту, выберите пункт «Export» из выпадающего меню (Рисунок 81).

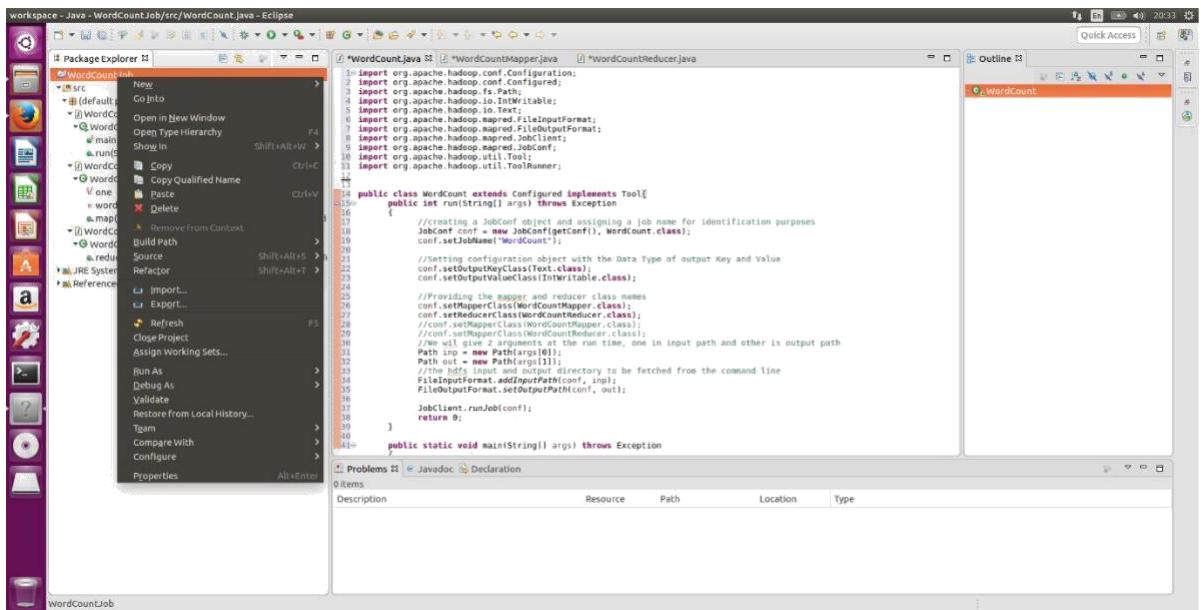


Рисунок 81 – вызов окна компоновки приложения в «jar» файл

В открывшемся окне экспорта выберите «Java» а затем «Runnable JAR file» (Рисунок 82).

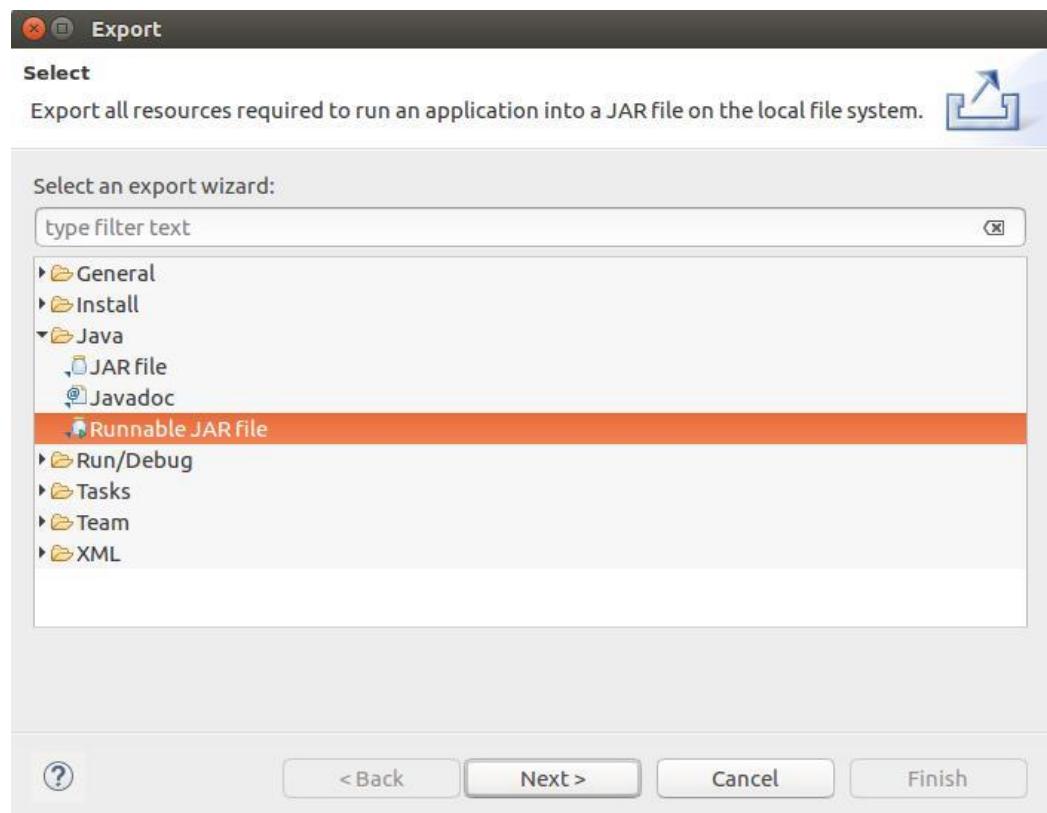


Рисунок 82 – окно экспорта приложения

В окне генерирования «jar» файла оставьте стандартный путь экспорта и выберите пункт «Extract required libraries into generated JAR», что позволит нам включить в наш файл все подключенные ранее библиотеки (Рисунок 83).

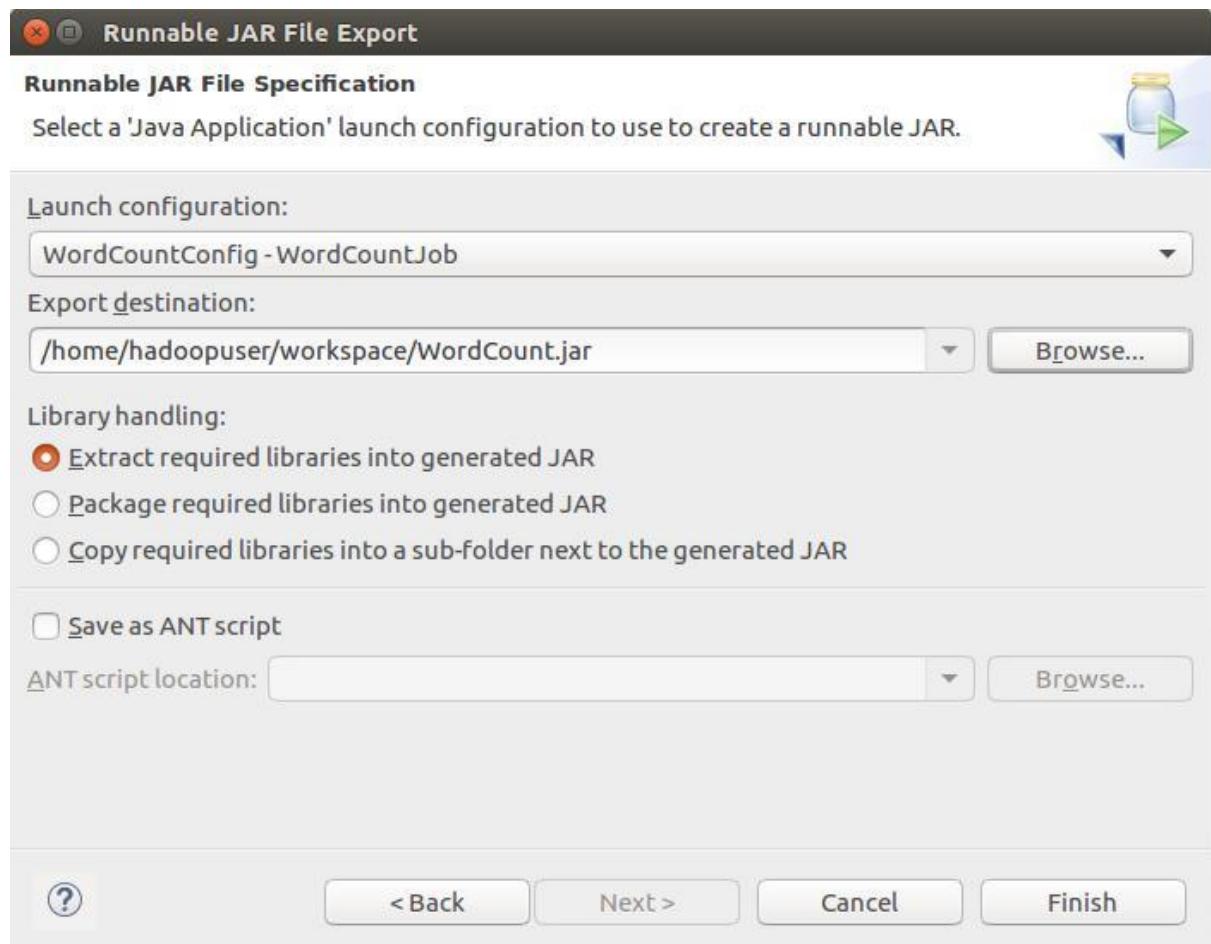


Рисунок 83 – настройки экспорта приложения

Наша программа готова. Мы также создали исполняем «jar» файл, который мы будем использовать для запуска программы.

2.2.3 ЗАПУСК ПРОГРАММЫ «WORDCOUNT»

Прежде чем мы запустим программу, нам нужно создать текстовый файл с текстом.

Данный файл имеет название «Data» и лежит в папке «DataTest» (Рисунок 84).

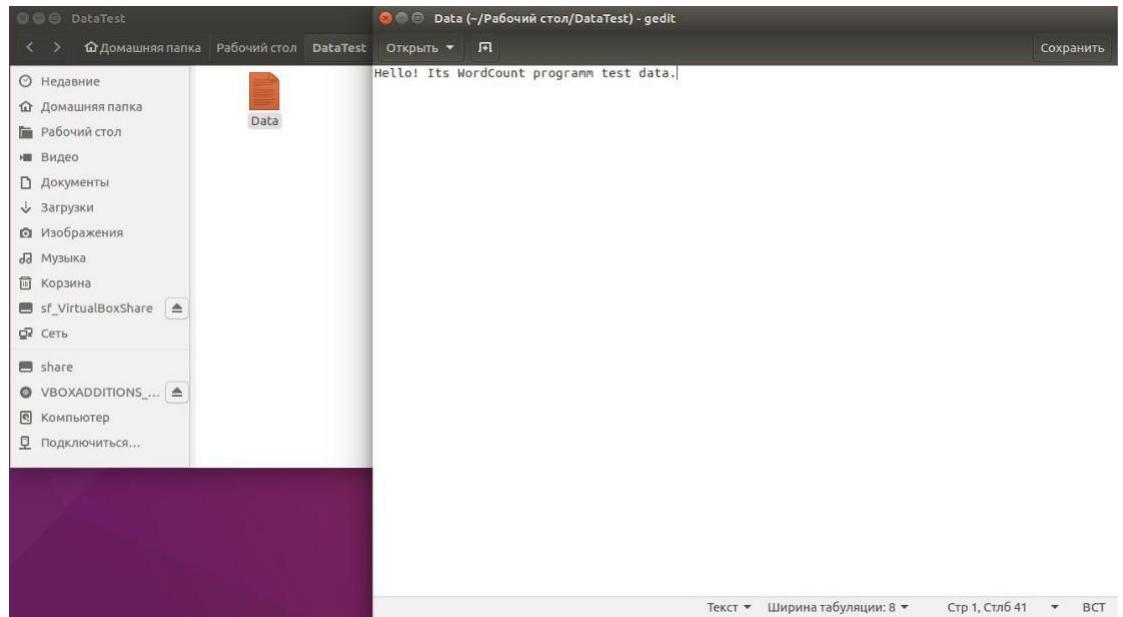


Рисунок 84 – текстовый файл

Теперь мы должны создать директории в «HDFS», где будет лежать наш текстовый файл. Для этого введите команды из рисунка 85.

```
hadoopuser@HadoopVM:~$ cd hadoop-2.6.0-cdh5.11.0/
hadoopuser@HadoopVM:~/hadoop-2.6.0-cdh5.11.0$ bin/hdfs dfs -mkdir /InputWordsTest
17/06/12 20:59:08 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
hadoopuser@HadoopVM:~/hadoop-2.6.0-cdh5.11.0$ bin/hdfs dfs -mkdir /InputWordsTest/TestData
17/06/12 20:59:18 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
hadoopuser@HadoopVM:~/hadoop-2.6.0-cdh5.11.0$
```

Рисунок 85 – команды создания нужных директорий

Данные команды выполняют следующие функции:

- «cd hadoop-2.6.0-cdh5.11.0/» осуществляет переход в корневую папку «Hadoop».
- «bin/hdfs dfs -mkdir /InputWordsTest» создает папку «InputWordsTest» в «HDFS».
- «bin/hdfs dfs -mkdir /InputWordsTest/TestData» создает папку «TestData» в «InputWordsTest».

Затем мы помещаем наш текстовый файл в созданную директорию с помощью команд с рисунка 86.

```
hadoopuser@HadoopVM:~/hadoop-2.6.0-cdh5.11.0$ bin/hdfs dfs -put '/home/hadoopuser/Рабочий%20стол/DataTest/Data' '/InputWordsTest/TestData'
17/06/12 21:01:11 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
hadoopuser@HadoopVM:~/hadoop-2.6.0-cdh5.11.0$
```

Рисунок 86 – команды копирования файла в «HDFS»

Проверяем наличие файла в папке «TestData» в «HDFS» через веб–браузер (Рисунок 87).

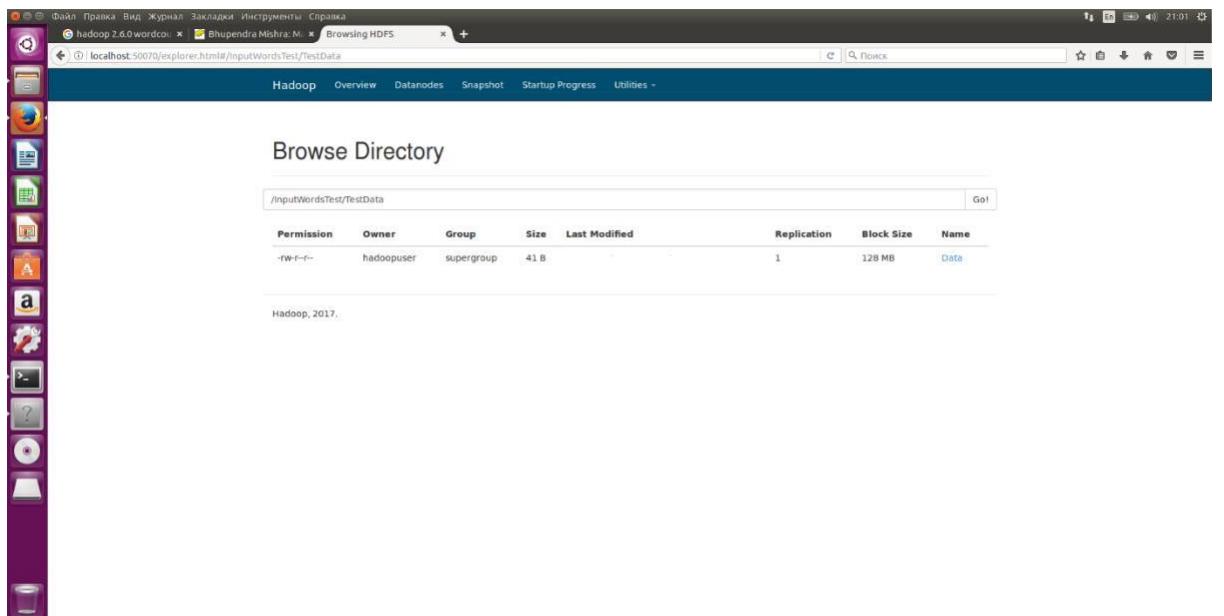


Рисунок 87 – проверка наличия файла в «HDFS»

Теперь мы можем запустить нашу программу. Для этого введите команду с рисунка 88.

```
hadoopuser@HadoopVM:~/hadoop-2.6.0-cdh5.11.0$ bin/hdfs dfs -put '/home/hadoopuser/Рабочий%20стол/DataTest/Data' '/InputWordsTest/TestData'
17/06/12 21:01:11 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
hadoopuser@HadoopVM:~/hadoop-2.6.0-cdh5.11.0$ bin/yarn jar '/home/hadoopuser/workspace/WordCount.jar' '/InputWordsTest/TestData' '/OutputWordsTest'
```

Рисунок 88 – команда запуска программы «WordCount»

Данная команда запускает нашу программу через фреймворк «YARN» и создает папку в «HDFS» под названием «OutputWordsTest» с результатом работы нашей программы.

Проверяем наличие результата работы программы через веб-браузер (Рисунок 89).

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	hadoopuser	supergroup	0 B	17/06/12 21:10:35	1	128 MB	_SUCCESS
-rw-r--r--	hadoopuser	supergroup	53 B	17/06/12 21:10:35	1	128 MB	part-r-00000

Рисунок 89 – проверка наличия результата работы программы

Теперь выводим результат с помощью команды с рисунка 90.

```
hadoopuser@HadoopVM:~/hadoop-2.6.0-cdh5.11.0$ bin/hdfs dfs -cat /OutputWordsTest/*
17/06/12 21:10:35 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Hello! 1
Its 1
wordCount 1
data. 1
programm 1
test 1
hadoopuser@HadoopVM:~/hadoop-2.6.0-cdh5.11.0$
```

Рисунок 90 – результат работы программы «WordCount»

Наша программа успешно подсчитала количество слов в нашем текстовом файле. Можно заметить, что она не распознает знаки препинания, поэтому имеет место улучшение нашей программы.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Tom White. Hadoop: The Definitive Guide: Storage and Analysis at internet Scale. -: "O'Reilly Media Inc. ", 2015. – 756 c.
2. Чак Лэм. Hadoop в действии. -: "Litres", 2017.
3. Srinath Perera. Hadoop Mapreduce Cookbook. -: "Packt Publishing Ltd", 2013. – 300 c.
4. VK Jain. Big Data and Hadoop. -: "Khanna Publishing", 2017. – 600 c.
5. Jason Venner, Sameer Wadkar, Madhu Siddalingaiah. Pro Apache Hadoop. -: "Apress", 2014. – 444 c.
6. Arun Murthy, Vinod Vavilapalli, Douglas Eadline, Joseph Niemiec, Jeff Markham. Apache Hadoop YARN: Moving beyond MapReduce and Batch Processing with Apache Hadoop 2. -: "Addison–Wesley Professional", 2014. – 400 c.
7. Dirk deRoos. Hadoop For Dummies. -: "John Wiley & Sons", 2014. – 408 c.
8. Vignesh Prajapati. Big Data Analytics with R and Hadoop. -: "Packt Publishing Ltd", 2013. – 238 c.
9. Boris Lublinsky, Kevin T. Smith, Alexey Yakubovich. Professional Hadoop Solutions. -: "John Wiley & Sons", 2013. – 504 c.
10. Kevin Sitto, Marshall Presser. Field Guide to Hadoop: An Introduction to Hadoop, Its Ecosystem, and Aligned Technologies. -: "O'Reilly Media, Inc.", 2015. – 132 c.
11. Garry Turkington. Hadoop Beginner's Guide. -: "Packt Publishing Ltd", 2013. – 398 c.
12. Danil Zburivsky. Hadoop Cluster Deployment. -: "Packt Publishing Ltd", 2013. – 126 c.

13. Kevin Roebuck. MapReduce: High-impact Strategies – What You Need to Know: Definitions, Adoptions, Impact, Benefits, Maturity, Vendors. –: "Lightning Source", 2011. – 170 c.
14. Donald Miner, Adam Shook. MapReduce Design Patterns: Building Effective Algorithms and Analytics for Hadoop and Other Systems. "O'Reilly Media, Inc.", 2012. – 250 c.
15. Thilina Gunarathne. Hadoop MapReduce v2 Cookbook – Second Edition. –: "Packt Publishing Ltd", 2015. – 322 c.
16. Билл Фрэнкс. Укрощение больших данных: Как извлекать знания из массивов информации с помощью глубокой аналитики. –: "Манн, Иванов и Фербер", 2014.
17. Виктор Майер–Шенбергер, Кеннет Кукиер. Большие данные: Революция, которая изменит то, как мы живем, работаем и мыслим. –: "Манн, Иванов и Фербер", 2013. – 240 c.
18. Peter Bühlmann, Petros Drineas, Michael Kane, Mark van der Laan. Handbook of Big Data. –: "CRC Press", 2016. – 464 c.
19. Frank J. Ohlhorst. Big Data Analytics: Turning Big Data into Big Money. –: "John Wiley & Sons", 2012. – 176 c.
20. Y. Lakshmi Prasad. Big Data Analytics Made Easy. –: "Notion Press", 2016. – 192 c.

ПРИЛОЖЕНИЕ А

Код класса «WordCount» на языке программирования «Java»:

```
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.FileInputFormat;
import org.apache.hadoop.mapred.FileOutputFormat;
import org.apache.hadoop.mapred.JobClient; import
org.apache.hadoop.mapred.JobConf; import
org.apache.hadoop.util.Tool; import
org.apache.hadoop.util.ToolRunner;
public class WordCount extends Configured implements
Tool{ public int run(String[] args) throws Exception {
//creating a JobConf object and assigning a job name for identification
purposes JobConf conf = new JobConf(getConf(), WordCount.class);
conf.setJobName("WordCount");
//Setting configuration object with the Data Type of output Key and Value
```

```

conf.setOutputKeyClass(Text.class);

conf.setOutputValueClass(IntWritable.class);

//Providing the mapper and reducer class names

conf.setMapperClass(WordCountMapper.class);

conf.setReducerClass(WordCountReducer.class);

//conf.setMapperClass(WordCountMapper.class);

//conf.setMapperClass(WordCountReducer.class);

//We wil give 2 arguments at the run time, one in input path and other is output
path

Path inp = new Path(args[0]);

Path out = new Path(args[1]);

//the hdfs input and output directory to be fetched from the command line

FileInputFormat.addInputPath(conf, inp);

FileOutputFormat.setOutputPath(conf, out);

JobClient.runJob(conf);

return 0;

}

public static void main(String[] args) throws Exception

{

// this main function will call run method defined above.

int res = ToolRunner.run(new Configuration(), new WordCount(),args);

```

```
System.exit(res);
```

```
}
```

```
}
```

ПРИЛОЖЕНИЕ Б

Код класса «WordCountMapper» на языке программирования «Java»:

```
import java.io.IOException;  
  
import java.util.StringTokenizer;  
  
import org.apache.hadoop.io.*;  
  
import org.apache.hadoop.mapred.*;  
  
public class WordCountMapper extends MapReduceBase implements  
Mapper<LongWritable, Text, Text, IntWritable>  
{  
  
    //hadoop supported data types  
  
    private final static IntWritable one = new  
    IntWritable(1); private Text word = new Text();  
  
    //map method that performs the tokenizer job and framing the initial key value  
pairs  
  
    // after all lines are converted into key–value pairs, reducer is called.  
  
    public void map(LongWritable key, Text value, OutputCollector<Text,  
IntWritable> output, Reporter reporter) throws IOException  
{  
  
    //taking one line at a time from input file and tokenizing the  
same String line = value.toString();  
  
    StringTokenizer tokenizer = new StringTokenizer(line);
```

```
//iterating through all the words available in that line and forming the key value
pair

while (tokenizer.hasMoreTokens())
{
    word.set(tokenizer.nextToken());
    //sending to output collector which inturn passes the same to reducer
    output.collect(word, one);
}

}
```

ПРИЛОЖЕНИЕ В

Код класса «WordCountReducer» на языке программирования «Java»:

```
import java.io.IOException;
import java.util.Iterator;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapred.*;
public class WordCountReducer extends MapReduceBase implements
Reducer<Text, IntWritable, Text, IntWritable>
{
    //reduce method accepts the Key Value pairs from mappers, do the aggregation
    based on keys and produce the final out put
    public void reduce(Text key, Iterator<IntWritable> values, OutputCollector<Text,
    IntWritable> output, Reporter reporter) throws IOException
    {
        int sum = 0;
        /*iterates through all the values available with a key and add them together and
        give the
        final result as the key and sum of its values*/
        while (values.hasNext())
        {
            sum += values.next().get();
        }
    }
}
```

```
    }  
  
    output.collect(key, new IntWritable(sum));  
  
}  
  
}
```