**Заявка №6 – Документ 2**

Hadoop. Текст.

**1 слайд**

****

Разработка была инициирована в начале 2005 с целью построения программной инфраструктуры распределённых вычислений для свободной программной поисковой машины на Java, её идейной основой стала публикация сотрудников Google о вычислительной концепции MapReduce.

Спустя годы разработок, начиная с 2010 года Hadoop неоднократно характеризуется как ключевая технология «больших данных», прогнозируется его широкое распространение для массово-параллельной обработки данных.

**2 слайд**

В итоге мы имеем такое определение Hadoop – это программный проект с открытым исходным кодом, предназначенный для эффективной обработки больших пакетов данных. Вместо одного большого компьютера для обработки и хранения данных Hadoop предлагает использовать для параллельного анализа огромных пакетов данных кластеры на базе стандартного аппаратного обеспечения.

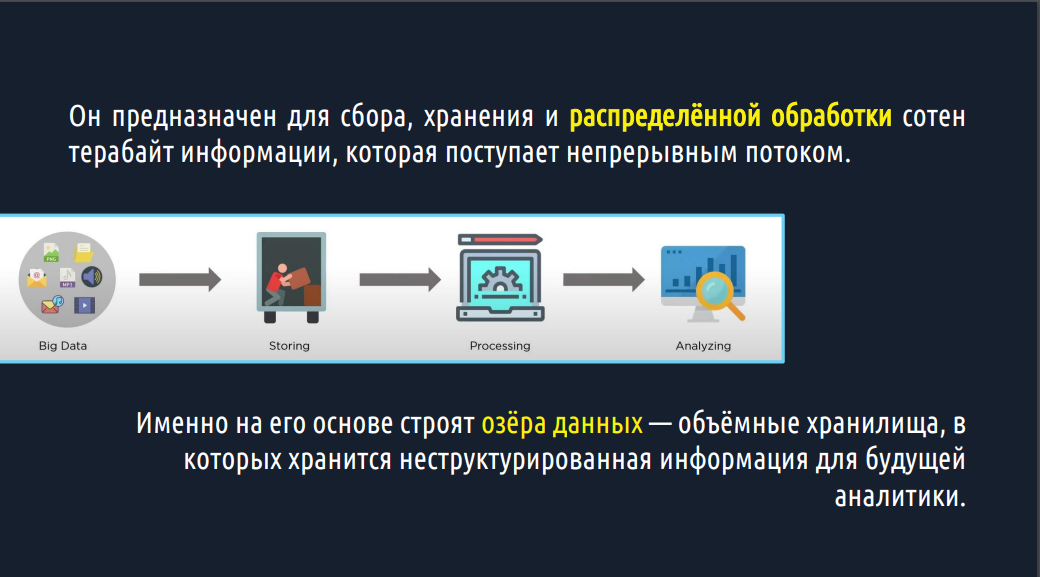


**3 слайд**

Apache Hadoop предназначен для обработки больших объемов данных в распределенной среде, используя фреймворк MapReduce. Он позволяет эффективно хранить и обрабатывать данные, которые могут быть распределены на несколько серверов в кластере. Hadoop также предоставляет множество инструментов для управления данными, таких как HDFS (Hadoop Distributed File System) для хранения больших объемов данных, Hive для работы с большими объемами структурированных данных, Hbase для работы с нереляционными данными и другие. Все эти возможности позволяют организациям обрабатывать и анализировать огромные объемы данных, которые ранее были неизвестны и недоступны.

Аналитика и доступ к большим данным

Hadoop отлично подходит для специалистов по обработке данных и инженеров машинного обучения, поскольку он позволяет нам выполнять расширенную аналитику для поиска закономерностей и разработки точных и эффективных моделей прогнозирования.



**4 слайд**

<https://www.codeinstinct.pro/2012/08/mapreduce-design.html>

Hadoop MapReduce использует архитектуру «master-worker», где master – единственный экземпляр управляющего процесса (JobTracker), как правило, запущенный на отдельной машине (вычислительном узле). Worker-процессы – это произвольное множество процессов TaskTracker, исполняющихся на DataNode.

JobTracker и TaskTracker «лежат» над уровнем хранения HDFS, и запускаются/исполняются в соответствии со следующими правилами:

экземпляр JobTracker исполняется на NameNode-узле HDFS;

экземпляры TaskTracker исполняются на DataNode-узле;

TaskTracker исполняются в соответствии с принципом «данные близко», т.е. процесс TaskTracker располагается топологически максимально близко с узлом DataNode, данные которого обрабатываются.

Взаимодействие JobTracker-узла с клиентом (программным) проходит по следующей схеме: JobTracker принимает задание (Job) от клиента и разбивает задание на множество M map-задач и множество R reduce-задач. Узел JobTracker использует информацию о файловых блоках (количество блоков и их месторасположение), расположенную в узле NamеNode, находящемуся локально, чтобы решить, сколько подчиненных задач необходимо создать на узлах типа TaskTracker. TaskTracker получает от JobTracker список задач (тасков), загружает код и выполняет его. Периодично TaskTracker отсылает JobTracker статус выполнения задачи.

Взаимодействия TaskTracker-узлов с программным клиентом отсутствуют.

По аналогии с архитектурой HDFS, где NameNode является единичной точкой отказа (Single point of failure), JobTracker также является таковой. Принцип восстановления в узлах JobTracker и TaskTracker описан ниже.

При сбое TaskTracker-узла JobTracker-узел переназначает задания неисправного узла другому узлу TaskTracker. В случае неисправности JobTracker-узла, для продолжения исполнения MapReduce-приложения, необходим перезапуск JobTracker-узла. При перезапуске узел JobTracker читает из специального журнала данные, о последней успешной контрольной точке (checkpoint), восстанавливает свое состояние на момент записи checkpoint и продолжает работу с места последней контрольной точки.



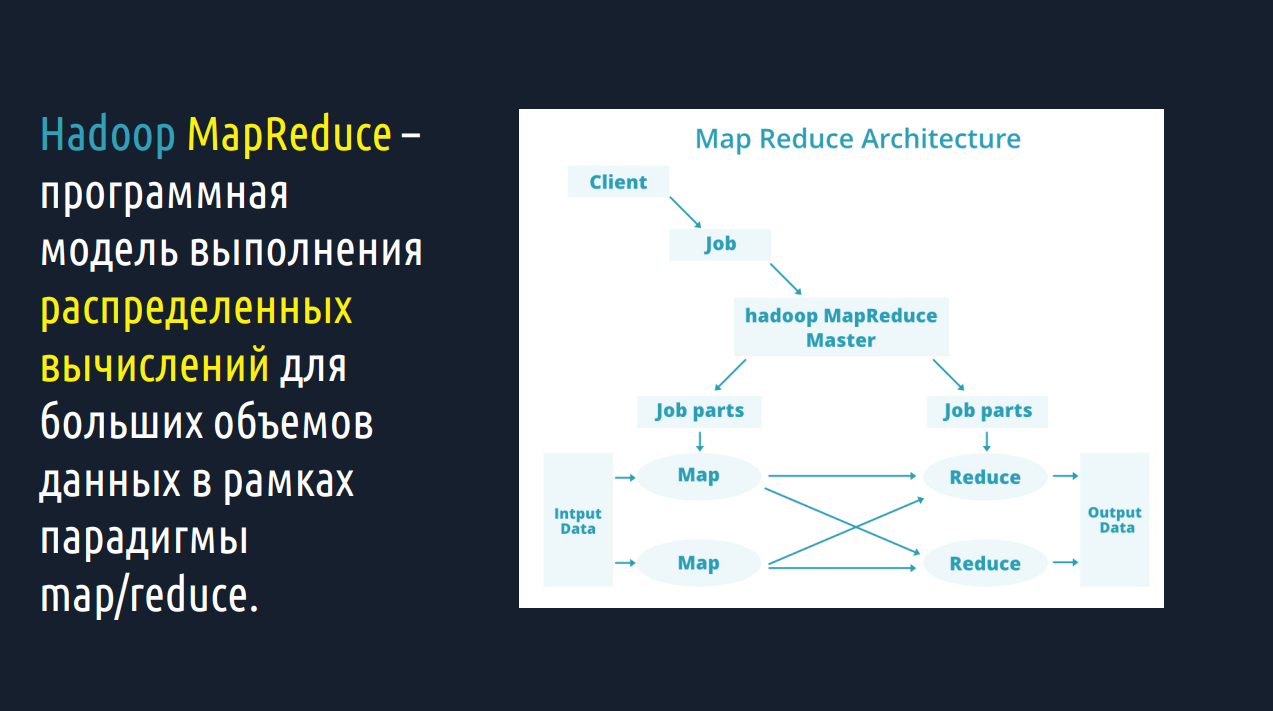
**5 слайд**

Apache Hadoop MapReduce — это программная платформа для создания заданий, обрабатывающих большие объемы данных. Входные данные разбиваются на независимые блоки, которые затем обрабатываются параллельно на узлах кластера. Задание MapReduce состоит из двух функций.

Mapper(Модуль сопоставления) — принимает входные данные, анализирует их (обычно с помощью фильтрации и сортировки) и передает кортежи (пары «ключ-значение»).

Reducer(Редуктор) — принимает кортежи, сформированные в модуле сопоставления, и выполняет операцию сводки, которая создает результат меньшего размера, объединяющий данные модуля сопоставления

**Разработчику приложения для Hadoop MapReduce необходимо реализовать базовый обработчик, который на каждом вычислительном узле кластера обеспечит преобразование исходных пар «ключ — значение» в промежуточный набор пар «ключ — значение»** (класс, реализующий интерфейс Mapper, назван по функции высшего порядка **Map**), **и обработчик, сводящий промежуточный набор пар в окончательный, сокращённый набор** (свёртку, класс, реализующий интерфейс **Reducer**). Каркас передаёт на вход свёртки отсортированные выводы от базовых обработчиков, сведе́ние состоит из трёх фаз — shuffle (тасовка, выделение нужной секции вывода), sort (сортировка, группировка по ключам выводов от распределителей — досортировка, требующаяся в случае, когда разные атомарные обработчики возвращают наборы с одинаковыми ключами, при этом, правила сортировки на этой фазе могут быть заданы программно и использовать какие-либо особенности внутренней структуры ключей) и собственно reduce (свёртка списка) — получения результирующего набора.



**6 слайд**

Тем не менее, считается, что горизонтальная масштабируемость в Hadoop-системах ограничена, для Hadoop до версии 2.0 максимально возможно оценивалась в 4 тыс. узлов при использовании 10 MapReduce-заданий на узел. Во многом этому ограничению способствовала концентрация в модуле MapReduce функций по контролю за жизненным циклом заданий, считается, что с выносом её в модуль YARN в Hadoop 2.0 и децентрализацией — распределением части функций по мониторингу на узлы обработки — горизонтальная масштабируемость повысилась.



**7 слайд**

Для решения "проблемы отстающего", в реализациях модели Map/Reduce используется техника спекулятивного выполнения. Она состоит в следующем: когда операции map или reduce завершились на большей части блоков данных, то копии задач на незавершившихся блоках запускаются на других узлах. Несмотря на дополнительные вычисления, такая техника часто ускоряет выполнение операций map и reduce на полном объеме данных. Техника спекулятивного выполнения подразумевает, что задачи map и reduce на разных блоках не обмениваются данными между собой в процессе исполнения.

Однако технология [MapR](https://www.bigdataschool.ru/wiki/mapr)educe хорошо подходит для задач распределенных вычислений в пакетном режиме, но из-за задержек (latency) не может использоваться для потоковой обработки в режиме реального времени.



**8 слайд**

Hadoop прозрачность достигается при помощи нескольких механизмов:

HDFS - это распределенная файловая система, которая обеспечивает прозрачный доступ к данным независимо от того, где они хранятся физически на узлах кластера. HDFS гарантирует доставку данных надежно и безопасно с повышенной степенью отказоустойчивости.

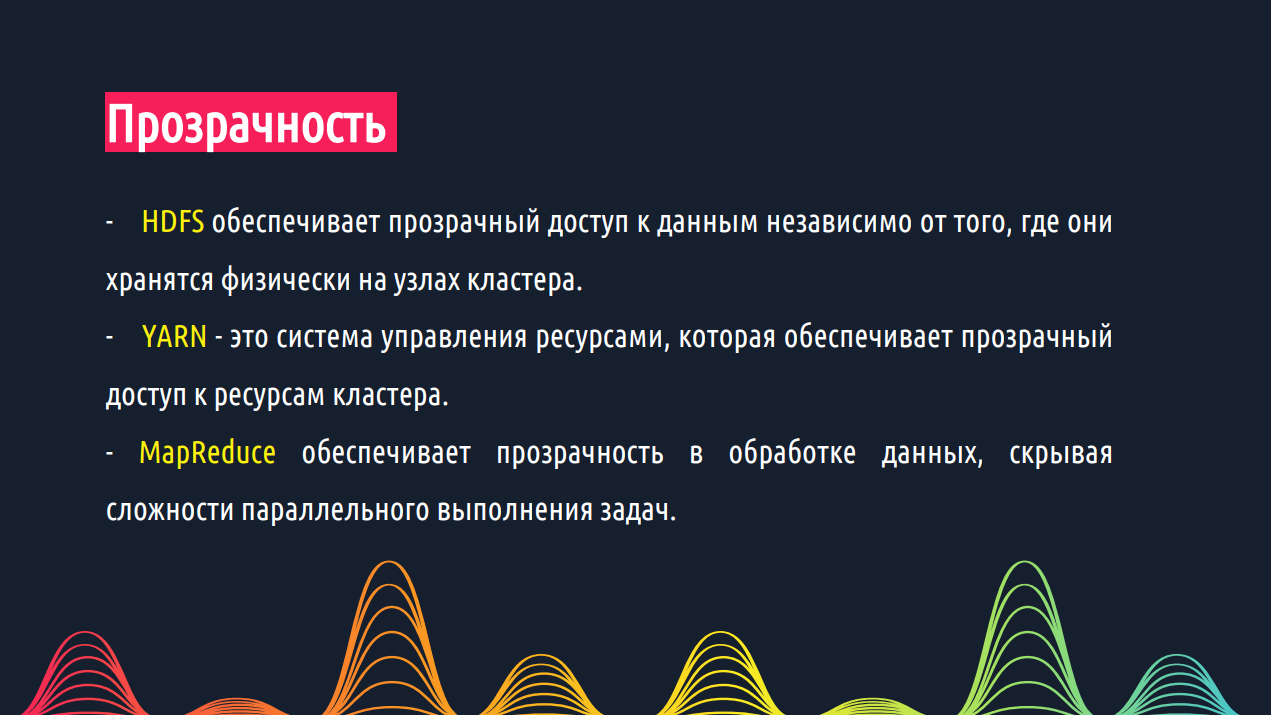
YARN - это система управления ресурсами, которая обеспечивает прозрачный доступ к ресурсам кластера. YARN позволяет выделять ресурсы для конкретных приложений и управлять ими независимо.

MapReduce - это программный интерфейс для распределенной обработки данных в Hadoop, который позволяет писать MapReduce задачи на языке Java, а затем запускать их на кластере Hadoop. MapReduce обеспечивает прозрачность в обработке данных, скрывая сложности параллельного выполнения задач.

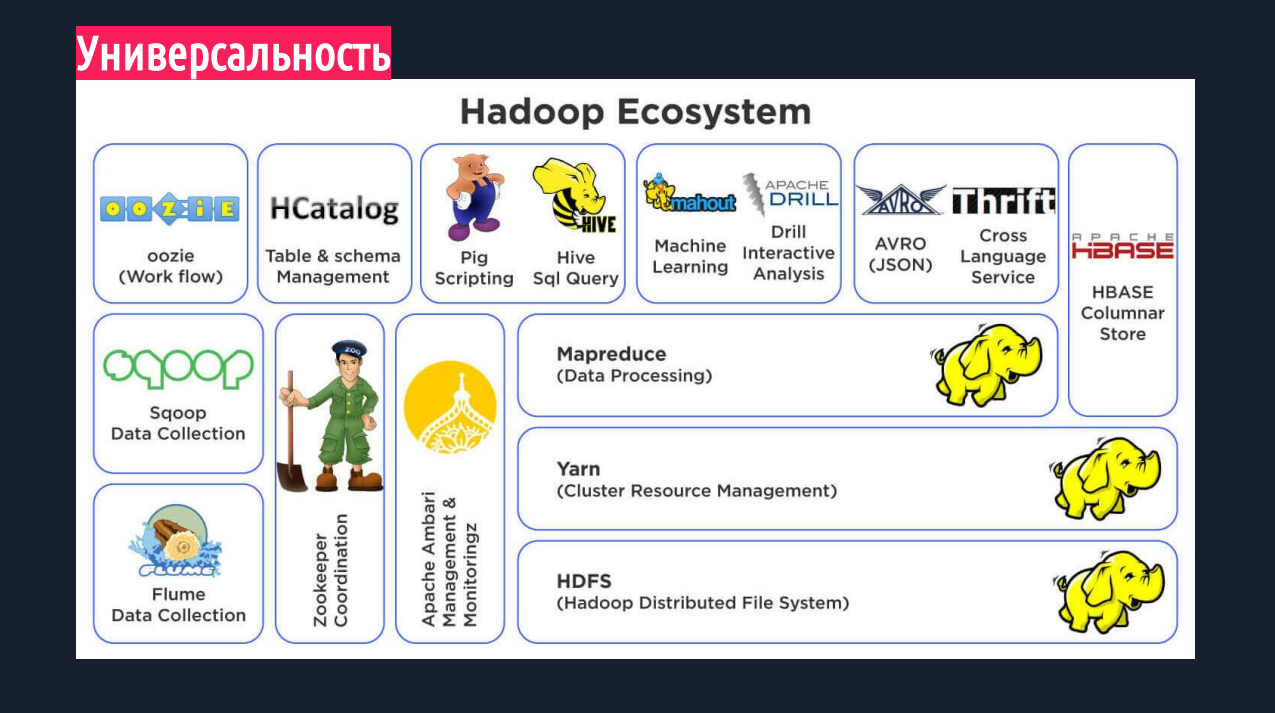
Apache Hive и Apache Pig - это инструменты для анализа данных в Hadoop, которые обеспечивают прозрачный доступ к данным, используя SQL-подобный язык и язык запросов на основе скриптов.

Apache HBase - это распределенная NoSQL база данных, которая обеспечивает прозрачный доступ к данным, используя ключ-значение и имеет поддержку транзакций.

В целом, Hadoop обеспечивает прозрачность, позволяя пользователям работать с данными, как будто они находятся в единственном месте, даже если они физически разделены в разных частях кластера.



**9 слайд**



**10 слайд**



**11 слайд**

Apache Hadoop имеет несколько интерфейсов взаимодействия для управления своими компонентами.

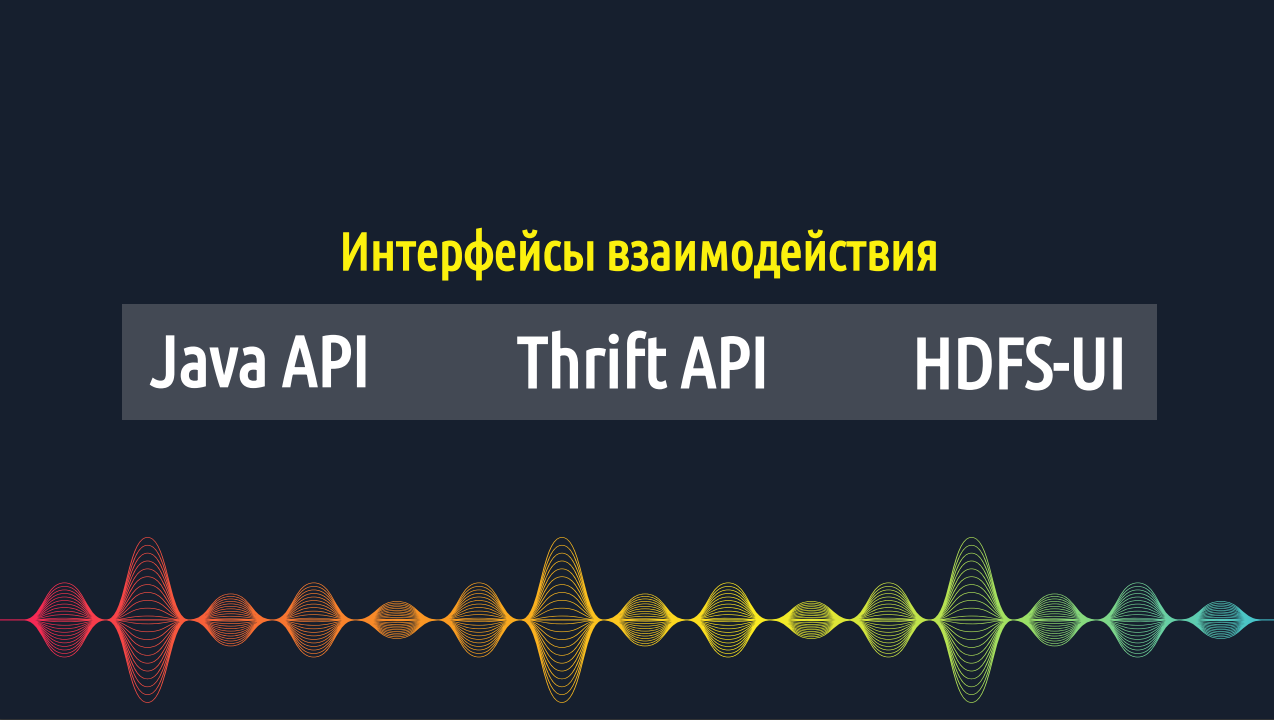
Command Line Interface (CLI) — командная строка Hadoop доступна через терминал и позволяет управлять кластером Hadoop, выполнять операции с файлами и запускать задачи MapReduce. Команды и аргументы CLI описаны в документации Hadoop.

Hadoop Web UI — графический интерфейс веб-приложения, доступный через браузер. Он позволяет просматривать статус кластера Hadoop, просматривать и управлять файлами и задачами MapReduce, просматривать Hadoop NameNode, DataNode и ResourceManager.

Hadoop API — это API Java, который используется для написания приложений, работающих с данными в Hadoop. Он позволяет создавать, читать и записывать данные в файлы Hadoop, а также выполнять задачи MapReduce.

Hadoop Streaming API — это API, который позволяет запускать процессы, написанные на других языках программирования, в Hadoop. Он использует стандартный ввод и вывод для взаимодействия с MapReduce.

Интерфейсы взаимодействия в Hadoop обеспечивают различные уровни абстракции для управления кластером Hadoop и выполнения задач обработки данных. Каждый интерфейс имеет свои преимущества и недостатки, и выбор определенного интерфейса взаимодействия зависит от конкретного варианта использования Hadoop.



**12 слайд**

**Стоимость**

Hadoop имеет открытый исходный код и использует недорогое товарное оборудование.

**Гибкость**

Hadoop разработан таким образом, что он может очень эффективно работать с любым набором данных, таким как структурированный (данные MySQL), полуструктурированный (XML, JSON), неструктурированный (изображения и видео).

**Скорость**

Hadoop использует распределенную файловую систему для управления своим хранилищем, то есть HDFS (распределенная файловая система Hadoop). В DFS (распределенная файловая система) файл большого размера разбивается на файловые блоки небольшого размера, которые затем распределяются между узлами, доступными в кластере Hadoop, поскольку это огромное количество файловых блоков обрабатывается параллельно, что делает Hadoop быстрее, из-за чего он обеспечивает высокий уровень производительности **по сравнению с традиционными системами управления базами данных**. Когда вы имеете дело с большим объемом неструктурированных данных, скорость является важным фактором, с Hadoop вы можете легко получить доступ к ТБАЙТ данных всего за несколько минут.

**Высокая пропускная способность**

Hadoop работает в распределенной файловой системе, где различные задания назначаются различным узлам данных в кластере, строка этих данных обрабатывается параллельно в кластере Hadoop, что обеспечивает высокую пропускную способность. Пропускная способность - это не что иное, как задача, выполняемая за единицу времени.

**Минимальный сетевой статус**

В Hadoop каждая задача разделена на различные небольшие подзадачи, которые затем назначаются каждому узлу данных, доступному в кластере Hadoop. Каждый узел данных обрабатывает небольшой объем данных, что приводит к низкому трафику в кластере Hadoop.

**Проблема с маленькими файлами**

Hadoop может эффективно работать с небольшим количеством файлов большого размера. Hadoop выходит из строя, когда ему требуется получить доступ к файлу небольшого размера в большом объеме. Это так много маленьких файлов, которые увеличивают Namenode и затрудняют работу.

**Уязвимость**

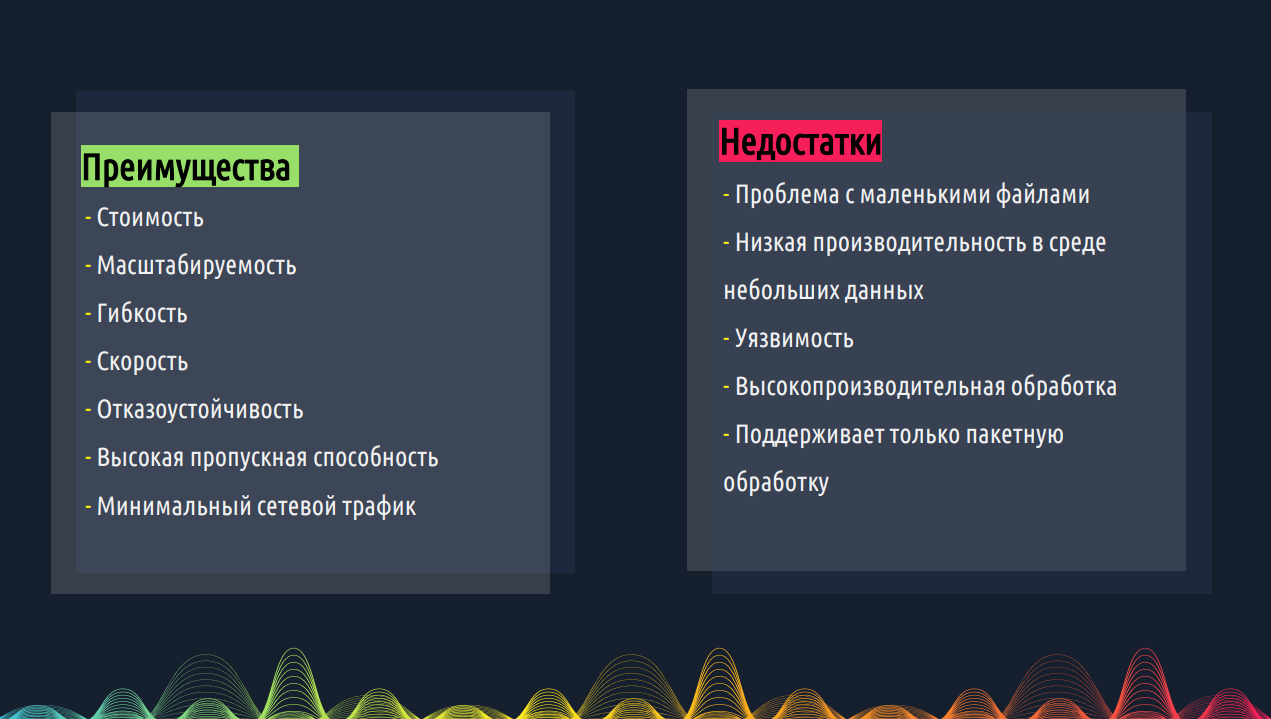
Hadoop - это фреймворк, написанный на Java, а Java является одним из наиболее часто используемых языков программирования, что делает его более небезопасным, поскольку он может быть легко использован любым киберпреступником.

**Высокопроизводительная обработка**

Операции чтения / записи в Hadoop являются неумеренными, поскольку мы имеем дело с данными большого размера, которые находятся в TB или PB. В Hadoop данные считываются или записываются с диска, что затрудняет выполнение вычислений в памяти и приводит к накладным расходам на обработку или высокой обработке.

**Поддерживает только пакетную обработку**

Пакетный процесс - это не что иное, как процессы, которые выполняются в фоновом режиме и не имеют никакого взаимодействия с пользователем. Механизмы, используемые для этих процессов внутри ядра Hadoop, не настолько эффективны. С его помощью невозможно выдавать выходные данные с низкой задержкой.



**13 слайд**

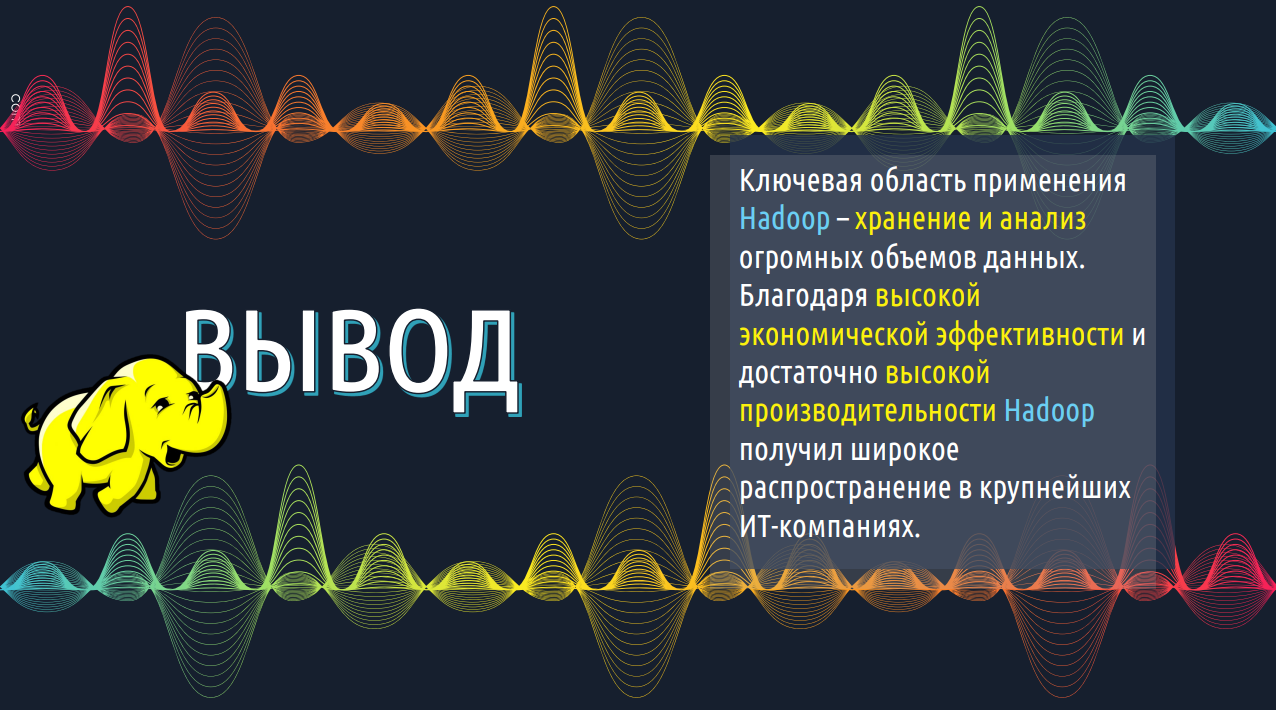
**Транспортные компании**: Используют Hadoop для сбора и анализа данных о транспортировке грузов, перемещении автомобилей, сроках доставки. Это помогает уменьшить расходы на топливо, прогнозировать лучшие маршруты, определять сроки технического обслуживания транспорта.

**Финансовые организации, в том числе банки:** Анализируют финансовую информацию и риски, выявляют мошеннические действия и разрабатывают защиту от них. Банки работают с большими объемами данных о клиентах и транзакциях, их анализ помогает предсказывать количество посетителей в отделениях, остатки средств в банкоматах, приток и отток корпоративных клиентов. Hadoop успешно справляется с такими задачами: по данным Syncsort, ⅔ организаций финансовой отрасли отмечают, что платформа делает бизнес более гибким и повышает операционную эффективность.

**Ритейлеры и продавцы услуг**: Собирают данные о продажах и транзакциях, поведении покупателей на сайте, информацию из соцсетей и с других ресурсов, финансовую информацию, отчетность об ассортименте и складских остатках. Зная, как ведут себя клиенты, можно делать персональные предложения и акции, предлагать востребованные товары, разрабатывать программы лояльности, повышать продажи.

****

**14 слайд**



**Случаи применения**

**Решения на основе данных**

Мы можем интегрировать структурированные и неструктурированные данные, которые не используются в хранилище данных или реляционной базе данных. Это позволяет нам принимать более точные решения, основанные на обширных данных.

**Аналитика и доступ к большим данным**

Hadoop отлично подходит для специалистов по обработке данных и инженеров машинного обучения, поскольку он позволяет нам выполнять расширенную аналитику для поиска закономерностей и разработки точных и эффективных моделей прогнозирования.

**Озера данных**

Решения для управления Hadoop могут помочь нам с интеграцией данных, безопасностью и качеством для озер данных.

Теперь, если вы посмотрите на все это и скажете: «Все хорошо, но это Java, и мы используем .NET», вам не о чем беспокоиться. Hadoop — это платформа, и вы можете привлекать клиентов к этой платформе на любом языке, который вам нужен. А поскольку Hadoop настолько хорош в качестве платформы распределенной обработки, он получает поддержку на всех языках.

В дополнение к ядру Hadoop я собираюсь представить систему Hive. Hive — это распределенная база данных SQL, которая располагается поверх Hadoop и HDFS. Есть три причины, почему стоит знать о Hive. Во-первых, потому что это значительно упрощает использование Hadoop и представляет его как технологию, которую вы можете использовать в своих проектах сегодня. Во-вторых, поскольку он использует SQL, технология, с которой большинство пользователей знакомо. В-третьих, потому что он используется Facebook для производства, а это значит, что он надежный, стабильный и в хорошем состоянии.

С Hadoop и Hive вы сможете размещать базы данных, которые содержат миллиарды записей, и выполнять запросы к ним (которые реализованы как задания Hadoop) в течение разумного времени. Это время будет регулироваться путем изменения размеров и рабочих характеристик машин в кластере.

Просто пару слов, чтобы вы успокоились по поводу всей проблемы кластера. Hadoop позволяет обслуживать и развертывать кластер машин, но для работы с ним нет необходимости иметь кластер машин. На самом деле подойдет любая старая одиночная машина. Вы можете запускать HDFS, Hadoop и Hive локально, и все работает отлично.

Давайте теперь углубимся в каждую из технологий.

# **Hadoop**

Hadoop - это набор программ с открытым исходным кодом, написанных на Java, которые можно использовать для выполнения операций с большим объемом данных. Hadoop - это масштабируемая, распределенная и отказоустойчивая экосистема. Основными компонентами Hadoop являются [6]:

* **Hadoop YARN**= управляет ресурсами системы и планирует их, разделяя рабочую нагрузку на кластер машин.
* **Распределенная файловая система Hadoop (HDFS)**= является кластерной системой хранения файлов, которая разработана для отказоустойчивости, предлагает высокую пропускную способность и высокую пропускную способность. Кроме того, он может хранить данные любого типа в любом возможном формате.
* **Hadoop MapReduce**= используется для загрузки данных из базы данных, их форматирования и количественного анализа.

Hadoop - это фреймворк для запуска приложений на больших кластерах, построенных из товарного оборудования. Hadoop прозрачно обеспечивает приложениям как надежность, так и движение. Hadoop реализует вычислительную парадигму под названием Map/Reduce, где приложение делится на множество небольших фрагментов работы, каждый из которых может быть выполнен или повторно выполняться на любом узле кластера. Кроме того, она предоставляет распределенную файловую систему (HDFS), которая хранит данные на вычислительных узлах, обеспечивая очень высокую совокупную пропускную способность по всему кластера. Как Map/Reduce, так и распределенная файловая система разработаны таким образом, что сбои узлов автоматически обрабатываются каркасом.

##### Блиц-обзор экосистемы: Hive, Pig, Oozie, Sqoop, Flume

В двух словах и обо всем.  
  
**Hive & Pig**. В большинстве случаев писать Map/Reduce job'ы на чистой Java — слишком трудоемкое и неподъемное занятие, имеющее смысл, как правило, лишь чтобы вытащить всю возможную производительность. Hive и Pig — два инструмента на этот случай. Hive любят в Facebook, Pig любят Yahoo. У Hive — SQL-подобный синтаксис ([сходства и отличия с SQL-92](http://hortonworks.com/blog/hive-cheat-sheet-for-sql-users/)). В лагерь Big Data перешло много людей с опытом в бизнес-анализе, а также DBA — для них Hive часто инструмент выбора. Pig фокусируется на ETL.  
  
**Oozie** — workflow-движок для jobs. Позволяет компоновать jobs на разных платформах: Java, Hive, Pig и т. д.  
  
Наконец, фреймворки, обеспечивающие непосредственно ввод данных в систему. Совсем коротко. **Sqoop** — интеграция со структурированными данными (РСУБД), **Flume** — с неструктурированными.

Hadoop - это фреймворк для распределенной обработки больших объемов данных, который изначально был разработан в компании Apache. В Hadoop прозрачность достигается при помощи нескольких механизмов:  
  
HDFS (Hadoop Distributed File System) - это распределенная файловая система, которая обеспечивает прозрачный доступ к данным независимо от того, где они хранятся физически на узлах кластера. HDFS гарантирует доставку данных надежно и безопасно с повышенной степенью отказоустойчивости.  
YARN (Yet Another Resource Manager) - это система управления ресурсами, которая обеспечивает прозрачный доступ к ресурсам кластера. YARN позволяет выделять ресурсы для конкретных приложений и управлять ими независимо.  
MapReduce - это программный интерфейс для распределенной обработки данных в Hadoop, который позволяет писать MapReduce задачи на языке Java, а затем запускать их на кластере Hadoop. MapReduce обеспечивает прозрачность в обработке данных, скрывая сложности параллельного выполнения задач.  
Apache Hive и Apache Pig - это инструменты для анализа данных в Hadoop, которые обеспечивают прозрачный доступ к данным, используя SQL-подобный язык и язык запросов на основе скриптов.  
Apache HBase - это распределенная NoSQL база данных, которая обеспечивает прозрачный доступ к данным, используя ключ-значение и имеет поддержку транзакций.  
В целом, Hadoop обеспечивает прозрачность, позволяя пользователям работать с данными, как будто они находятся в единственном месте, даже если они физически разделены в разных частях кластера.