



НBase

Иван Пузыревский
Разработчик Яндекс

Презентация: Ивченко Олег, 094а

Москва, 2015 (Апрель)

План лекции

- Введение
- Архитектура BigTable-подобных систем
- Методы чтения и записи
- Обеспечение надёжности и отказоустойчивости
- Best practices

HDFS

- Особенности HDFS:
 - Пишем **большими** блоками (≈ 64 Mb),
 - Читаем в режиме потока,
 - Обращаемся к системе сравнительно **редко**.

Системы доступа по ключу

- Примеры:
 - Рекламные, рекомендательные системы (ключ – профиль пользователя)
 - Почта (по ключу хранится список писем)
 - Задачи сбора статистики

Системы доступа по ключу

- Проблемы при использовании HDFS:
 - Читаем и пишем *небольшие* данные ключу (100 bytes – 1 Mb)
 - *Часто* обращаемся к системе ($\approx 10^4$ раз/с)

Системы доступа по ключу

- Проблемы при использовании HDFS:
 - Читаем и пишем *небольшие* данные ключу (100 bytes – 1 Mb)
 - *Часто* обращаемся к системе ($\approx 10^4$ раз/с)
- Решения:
 - BigTable (Google, 2006) -> **HBase**
 - Dynamo (Amazon, 2007) -> **Apache Cassandra**

Развитие систем работы с данными

- Google
- Hadoop ecosystem

A solid blue horizontal rectangular bar.

GFS

A solid green horizontal rectangular bar.

HDFS

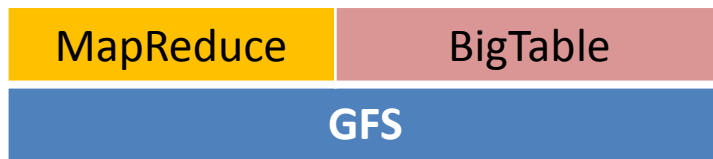
Развитие систем работы с данными

- Google
- Hadoop ecosystem

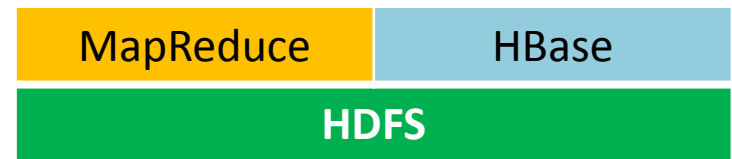


Развитие систем работы с данными

- Google

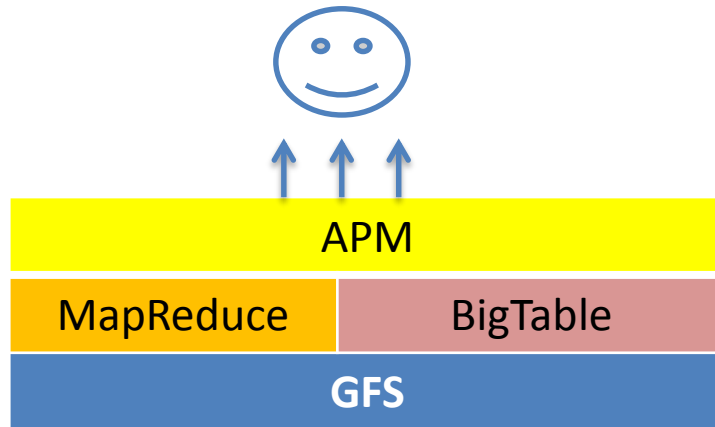


- Hadoop ecosystem

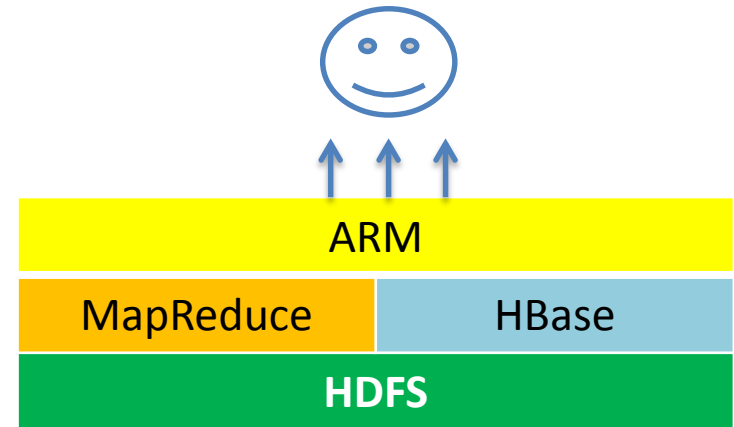


Развитие систем работы с данными

- Google



- Hadoop ecosystem



Распределение данных по машинам

- С помощью hash-функции

$$i = \text{hash}(k) \% M$$

- Равномерное

$$i = [k/M]$$

Распределение данных по машинам

- С помощью hash-функции

$$i = \text{hash}(k) \% M$$

– Подряд идущ. ключи находятся в разных местах

- Равномерное

$$i = [k/M]$$

– Порядок ключей не нарушается

– Приспособлено для сканирования диапазона между двумя ключами

Распределение данных по машинам

- С помощью hash-функции

$$i = \text{hash}(k) \% M$$

– Подряд идущ. ключи находятся в разных местах

Реализация: **Dynamo**

- Равномерное

$$i = [k/M]$$

– Порядок ключей не нарушается

– Приспособлено для сканирования диапазона между двумя ключами

Реализация: **Hbase**

План лекции

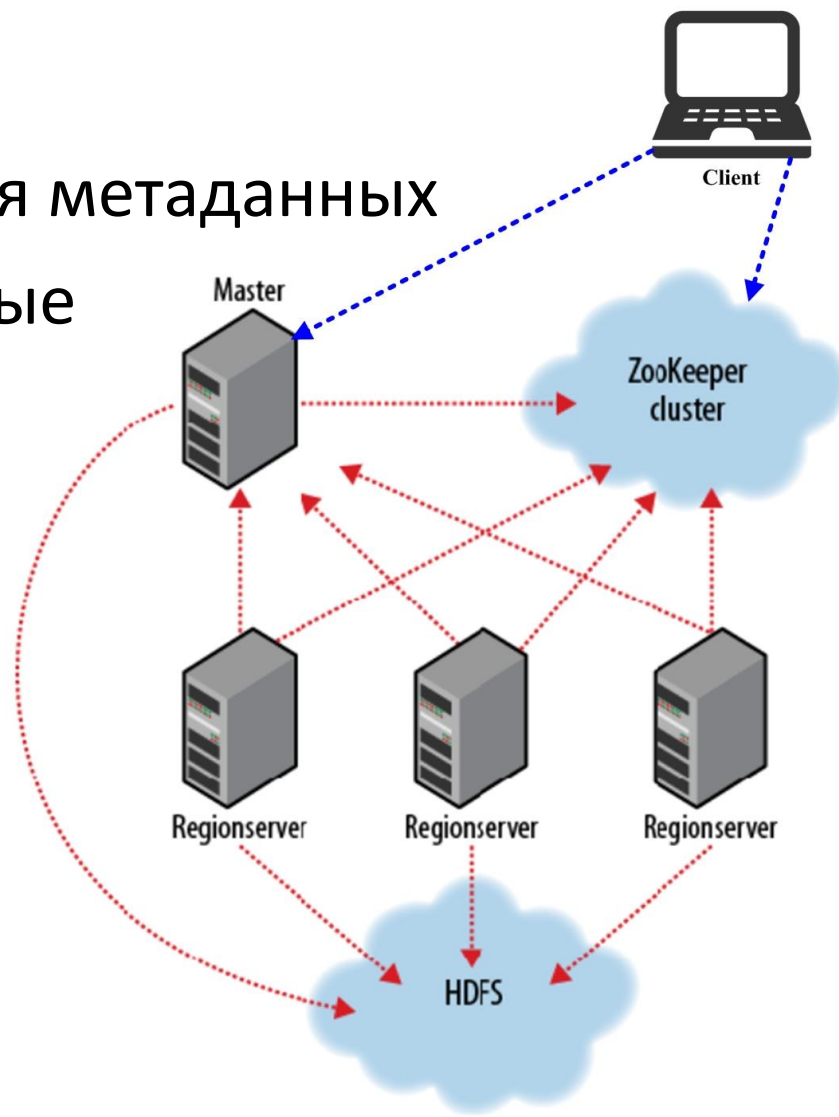
- Введение
- **Архитектура BigTable-подобных систем**
- Методы чтения и записи
- Обеспечение надёжности и отказоустойчивости
- Best practices

Архитектура HBase

Пространство ключей режется на части – tablets.

- **Master server**

- контролирует изменения метаданных
- перераспределяет данные между машинами



Архитектура HBase

Пространство ключей режется на части – tablets.

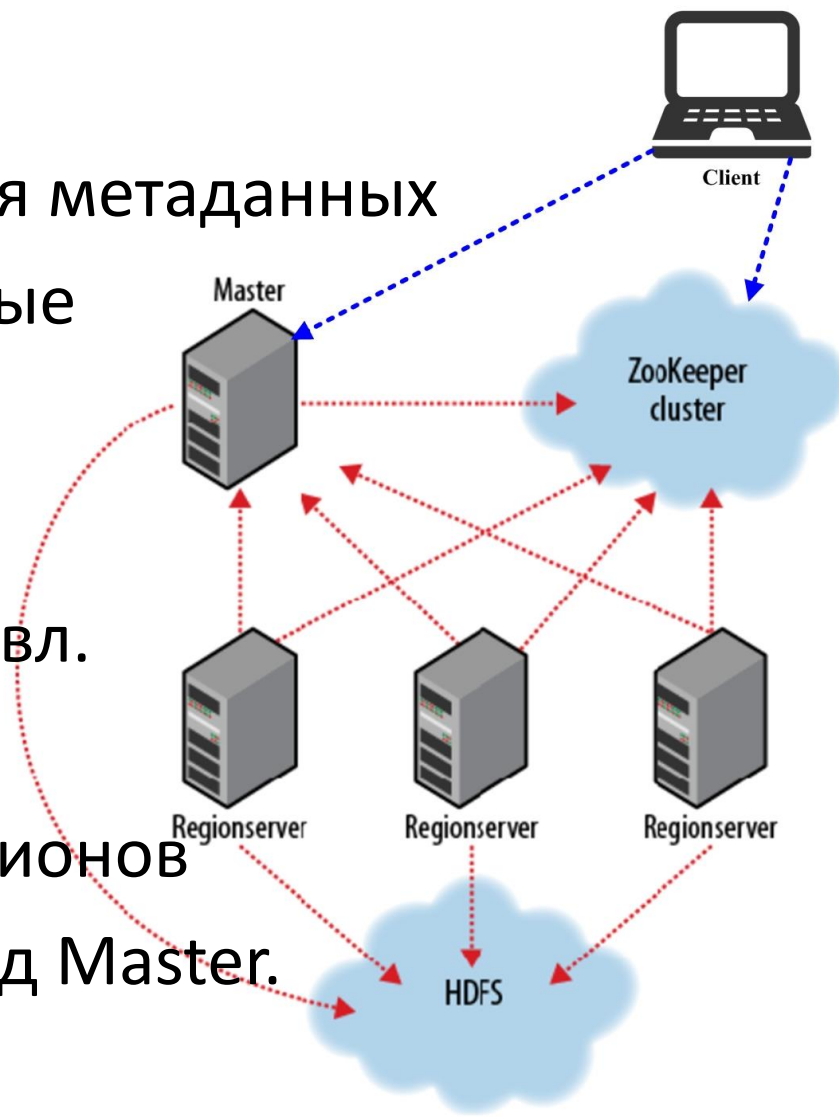
- **Master server**

- контролирует изменения метаданных
- перераспределяет данные между машинами

- **Region (tablet) server**

- хранит tablets и осуществл. операции с ним
- содержит несколько регионов

NB: доступ клиента в обход Master.



План лекции

- Введение
- Архитектура BigTable-подобных систем
- **Методы чтения и записи**
- Обеспечение надёжности и отказоустойчивости
- Best practices

Запись данных

- За 1 операцию записи отвечает 1 region server

Запись данных

- За 1 операцию записи отвечает 1 region server
- Предполагаем, что данные хранятся в виде таблицы (key, value).
 - составной ключ $key = (row_key, col_key, time)$.
 - time – время записи чтобы избежать коллизий
 - возможность читать не всю информацию, а интересующий фрагмент
 - возможность настраивать политики хранения

Запись данных

- За 1 операцию записи отвечает 1 region server
 - Предполагаем, что данные хранятся в виде таблицы (key, value).
 - составной ключ $key = (row_key, coll_key, time)$.
 - time – время записи чтобы избежать коллизий
 - возможность читать не всю информацию, а интересующий фрагмент
 - возможность настраивать политики хранения
- Пример:** row_key = userId, coll_key – более конкретная информация (пол, возраст etc.)

Запись данных

- За 1 операцию записи отвечает 1 region server
- Предполагаем, что данные хранятся в виде таблицы (key, value).
 - составной ключ $key = (row_key, coll_key, time)$.
 - time – время записи чтобы избежать коллизий
 - возможность читать не всю информацию, а интересующий фрагмент
 - возможность настраивать политики хранения
- *Пример:* row_key = userId, coll_key – более конкретная информация (пол, возраст etc.)
- Сортировка лексикографически по умолчанию



Запись на 1 машине

- Каждая итерация записи – работа с HDD?
 - Долго. 1 операция $\approx k$ мс.
- Выход?



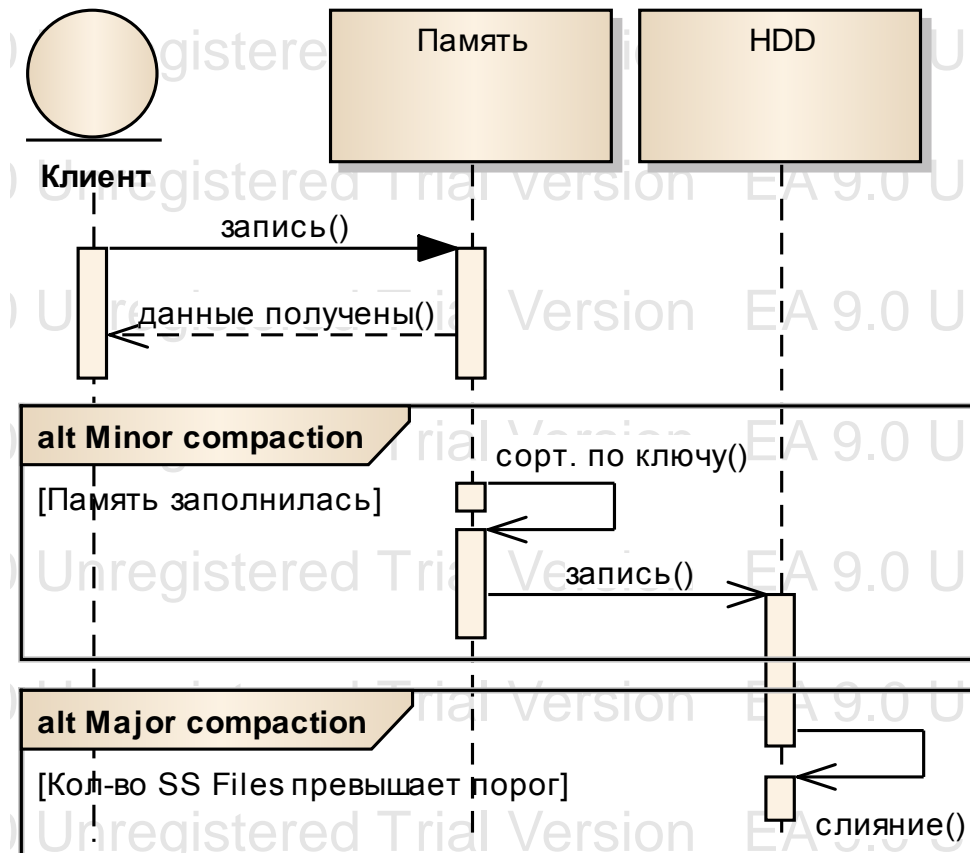
Запись на 1 машине

- Каждая итерация записи – работа с HDD?
 - Долго. 1 операция $\approx k$ мс.
- Выход:
 - писать в оперативную память
 - постоянная запись на диск



Запись на 1 машине

- Реализация в HBase: сохранение в память + 2 фоновые процедуры.
 - **Minor compaction** – сохранение памяти на диск в отсортиров. виде (в BigTable **SS Files**, в HBase - **HFiles**)
 - **Major compaction**:
 - объединение многих SS Files в один
 - удаление перезаписей значений одного ключа
 - удаление данных



Удаление данных

1. По данному ключу пишется маркер.

Удаление данных

1. По данному ключу пишется маркер.
2. В момент major compaction данные и ключ удаляются.

Major compaction

- Как оптимизировать?
 - мало SS Files – хорошо для чтения
 - мало операций перезаписи SS Files – для записи

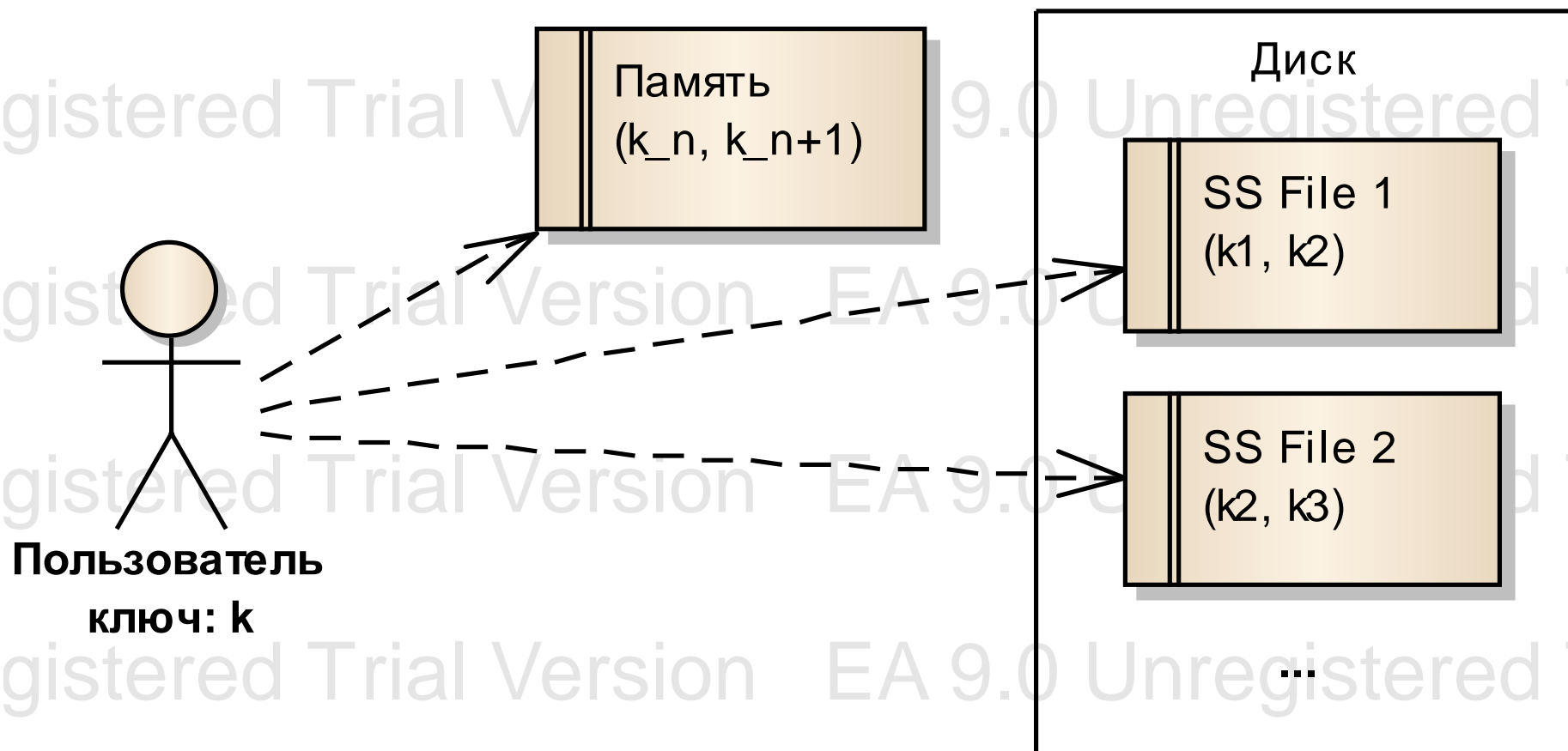
Major compaction

- Как оптимизировать?
 - мало SS Files – хорошо для чтения
 - мало операций перезаписи SS Files – для записи
- Решение:
 - не допускать слишком большой загрузки одной машины
 - упорядочивать данные в регионах



Чтение на 1 машине

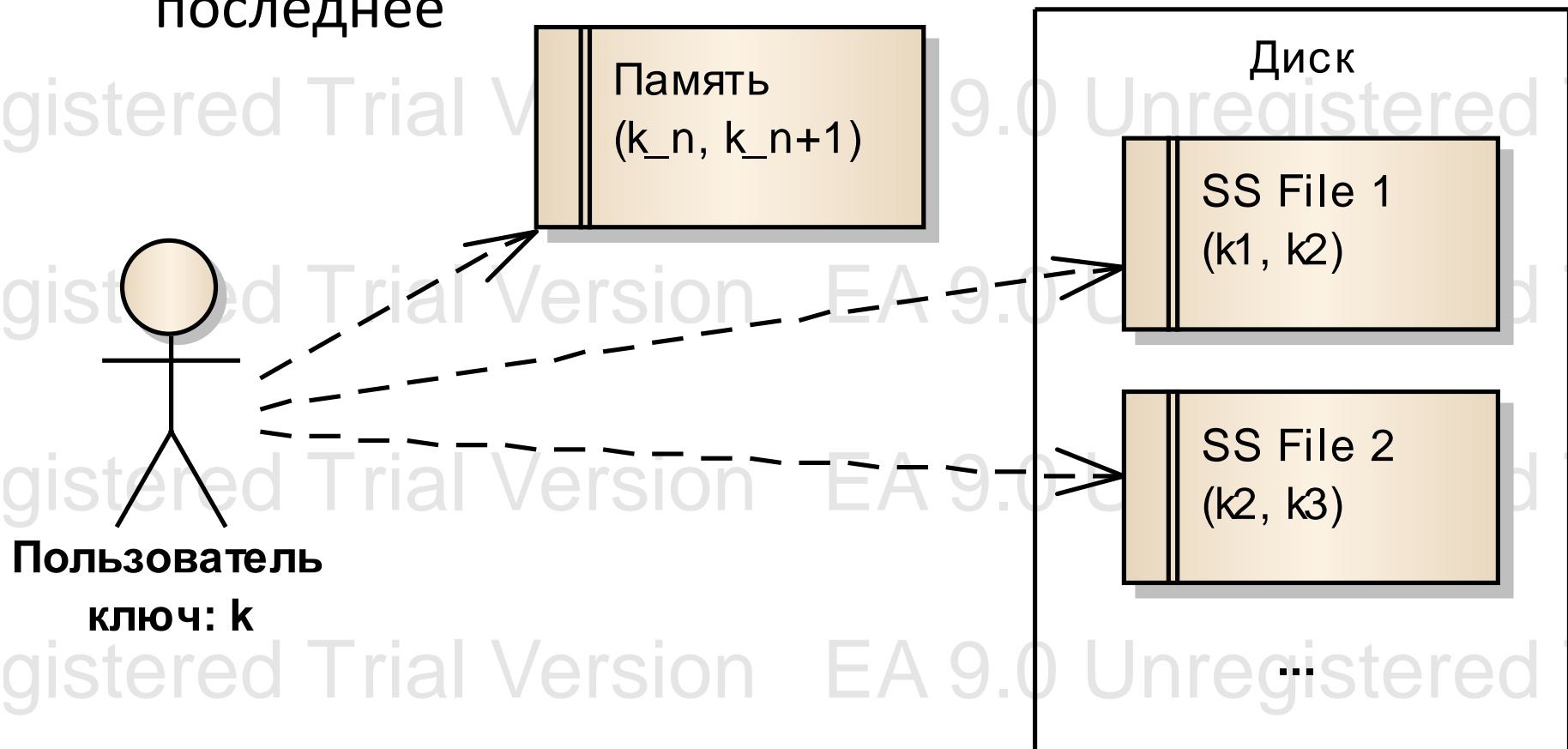
- Имеется ключ k , хотим получить данные





Чтение на 1 машине

- Имеется ключ k , хотим получить данные
 - если значений по ключу k несколько возвращаем последнее



Чтение данных

1. Master ищет по ключу нужную машину.
2. Машина проверят, есть ли у неё необходимые SS Files.
3. Актуализирует данные.

Чтение данных

1. Master ищет по ключу нужную машину.
 2. Машина проверят, есть ли у неё необходимые SS Files.
 3. Актуализирует данные.
- SS Files хранятся в HDFS
 - Машины делают реплики себе

Ещё о Master Server

Master выдаёт диапазон ключей при создании
нов. таблицы

Ещё о Master Server

Master выдаёт диапазон ключей при создании нов. таблицы

Функции Master'а в исключит. ситуациях:

- Регион очень большой
 - старая машина обслуживает только половину
 - новая машина получает в распоряжение 2-ю половину

Ещё о Master Server

Master выдаёт диапазон ключей при создании нов. таблицы

Функции Master'а в исключит. ситуациях:

- Регион очень большой
 - старая машина обслуживает только половину
 - новая машина получает в распоряжение 2-ю половину
- Регион очень маленький
 - сливаются 2 соседних региона

План лекции

- Введение
- Архитектура BigTable-подобных систем
- Методы чтения и записи
- **Обеспечение надёжности и отказоустойчивости**
- Best practices

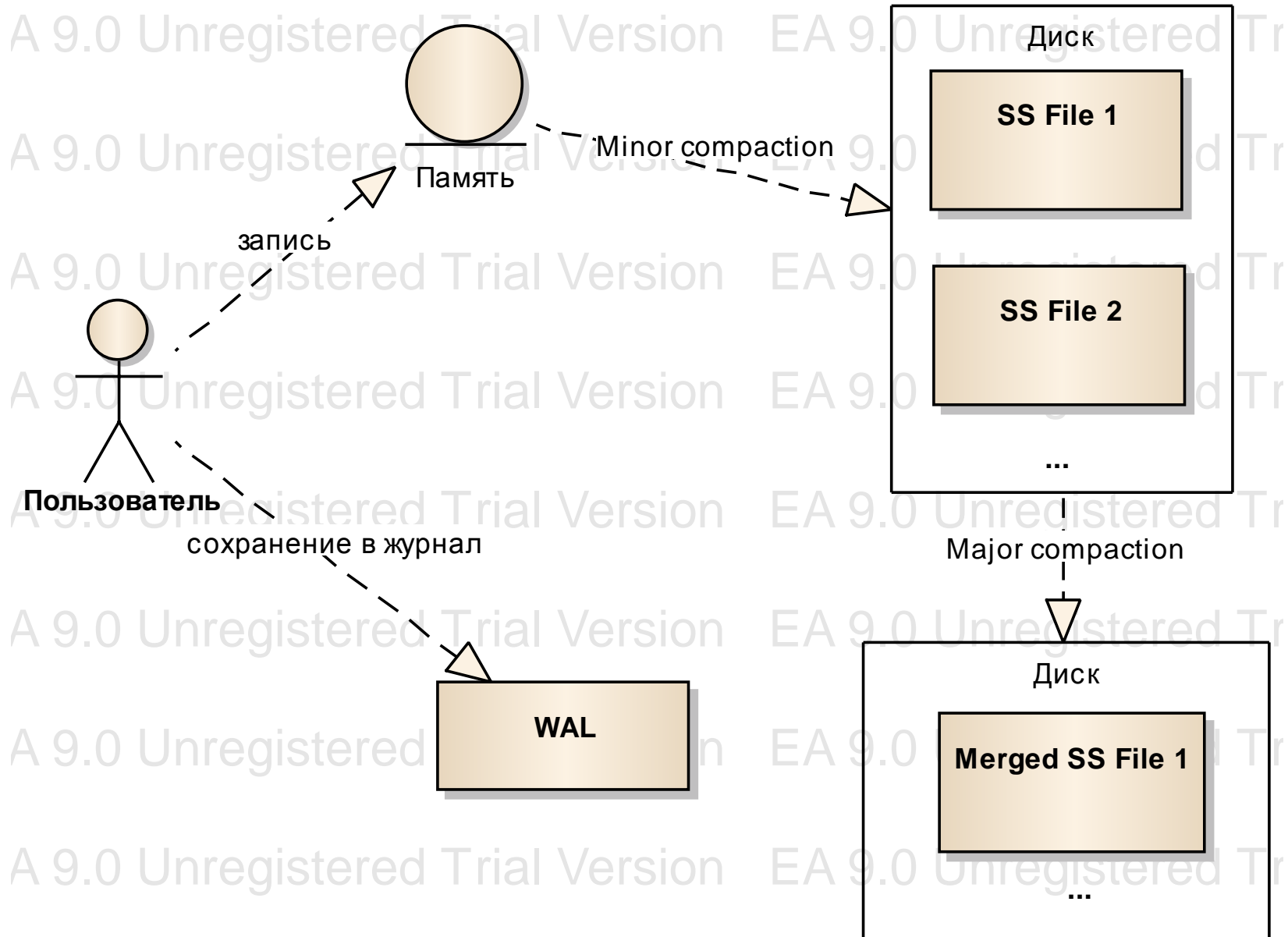
Отказоустойчивость

- Записали данные в память, но ещё не скинули на диск.
 - Если произойдёт сбой, данные потеряются
- Решение ?

Отказоустойчивость

- Записали данные в память, но ещё не скинули на диск.
 - Если произойдёт сбой, данные потеряются
- Решение:
 - Write-ahead log – журнал операций записи
 - Обычно на каждый регион по 1 WAL
 - В больших кластерах 1 WAL обслуживает 1 Region Server

Отказоустойчивость



План лекции

- Введение
- Архитектура BigTable-подобных систем
- Методы чтения и записи
- Обеспечение надёжности и отказоустойчивости
- Best practices

Best practices

- Нужно ограничить Java heap size (≈ 12 Gb) чтобы избежать задержки из-за Garbage collector.
- Настроить сжатие (LZO, Snappy)
- Не перегружать (не создавать слишком много регионов) на 1 машине

Пользователи HBase



YAHOO!

ebay box



AdRoll
FLURRY

explorys appnexus



photobucket

KLOUT OP@WER

AOL

Gravity

Pinterest

GROUPON



TREND
MICRO

rocketfuel
Artificial intelligence. Real results.