Лекция 4. Hadoop MapReduce

Основные темы

- Пример MapReduce
- Hadoop MapReduce
- Развертывание приложения MapReduce
- Перетасовка (Shuffling) MapReduce
- Отказоустойчивость

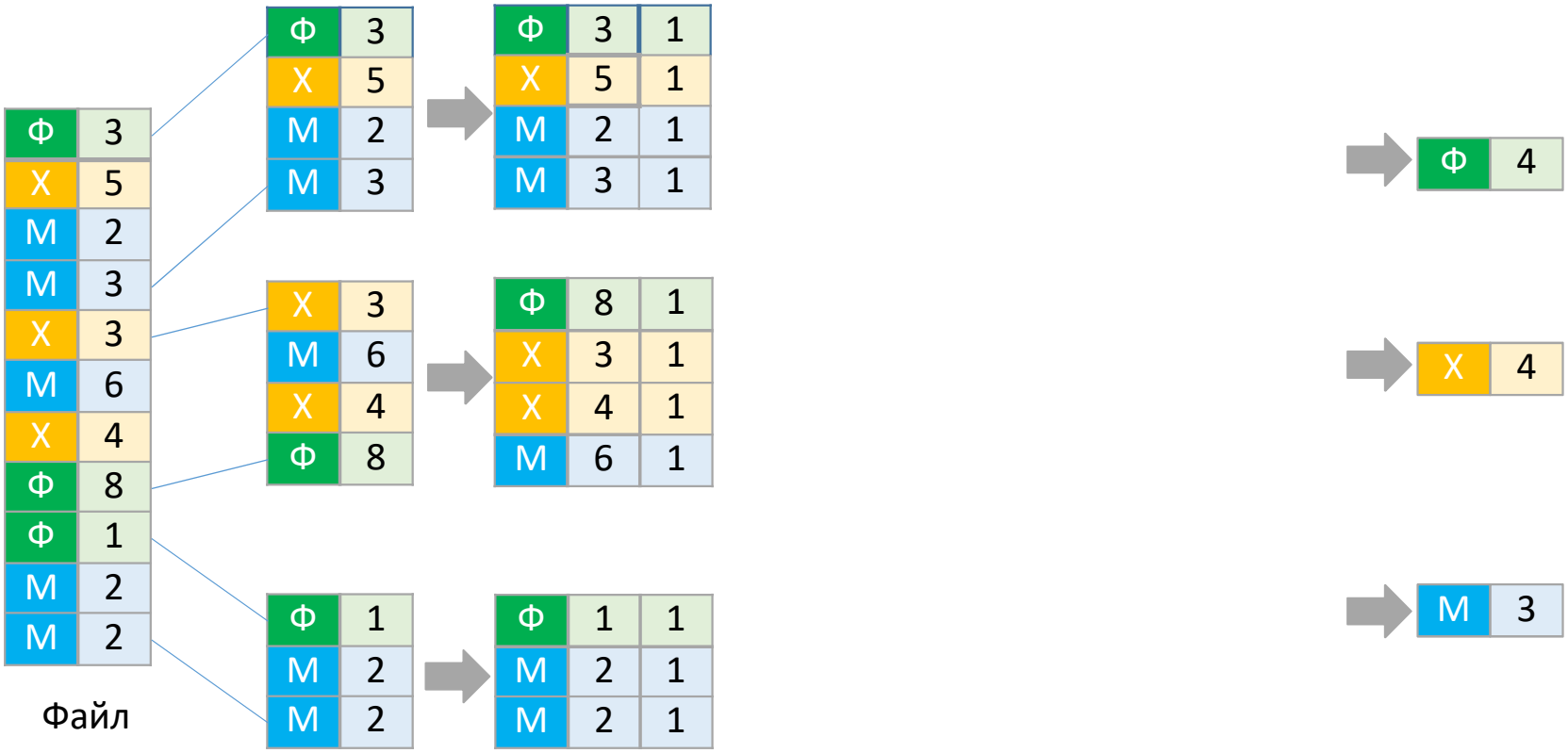
Среднее значение

Ф	3
Х	5
М	2
М	3
Х	3
М	6
Х	4
Ф	8
Ф	1
М	2
М	2

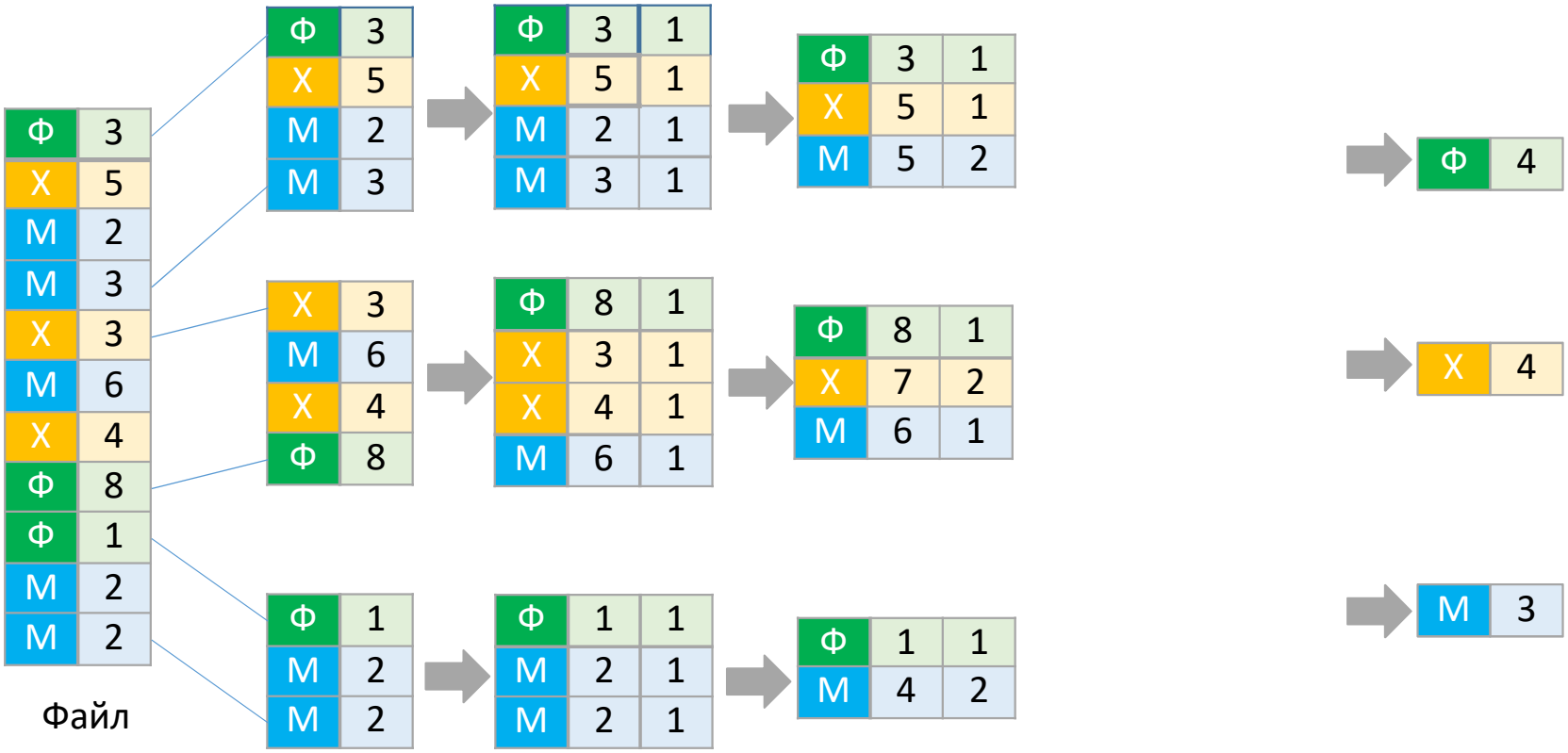
Файл

Среднее значение

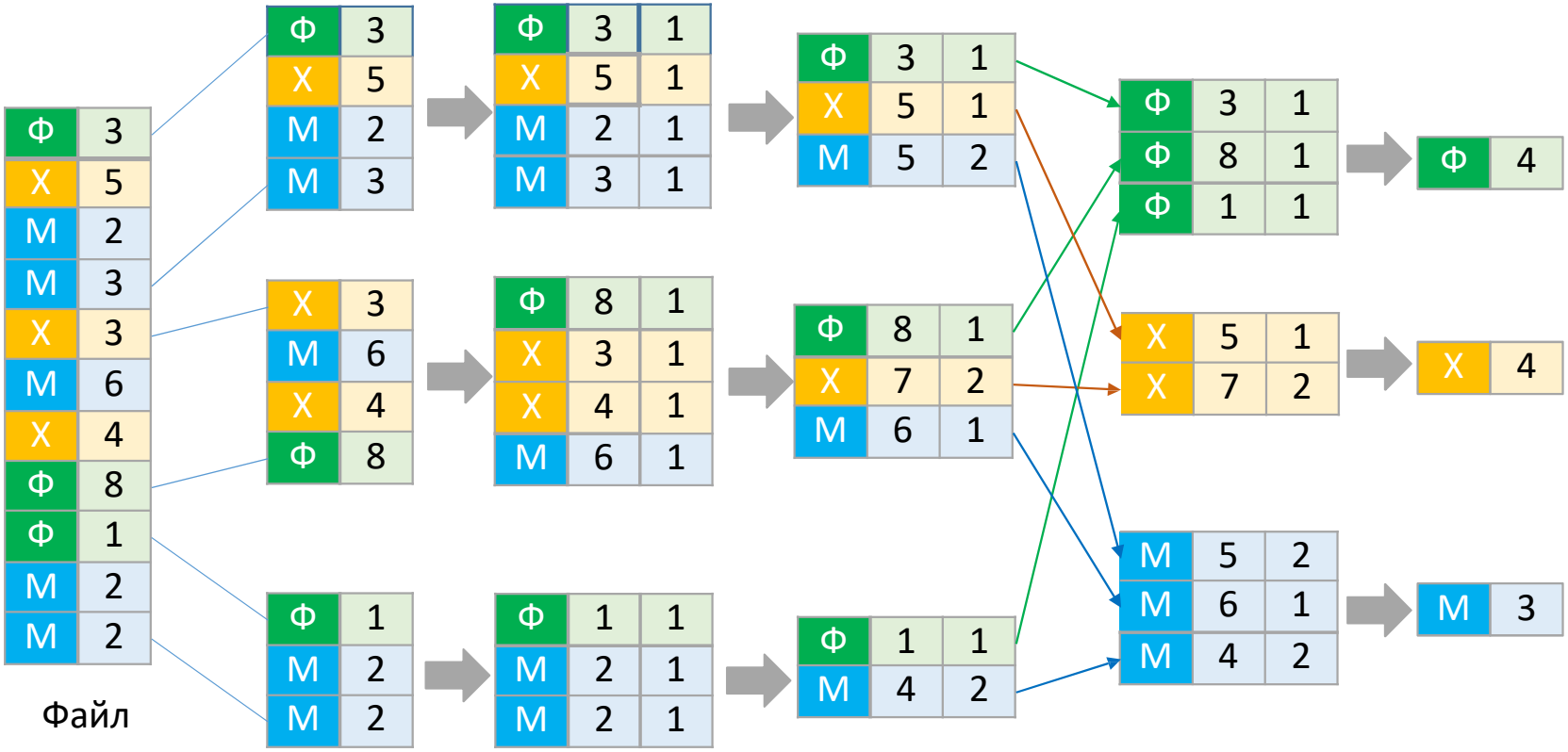
Среднее значение



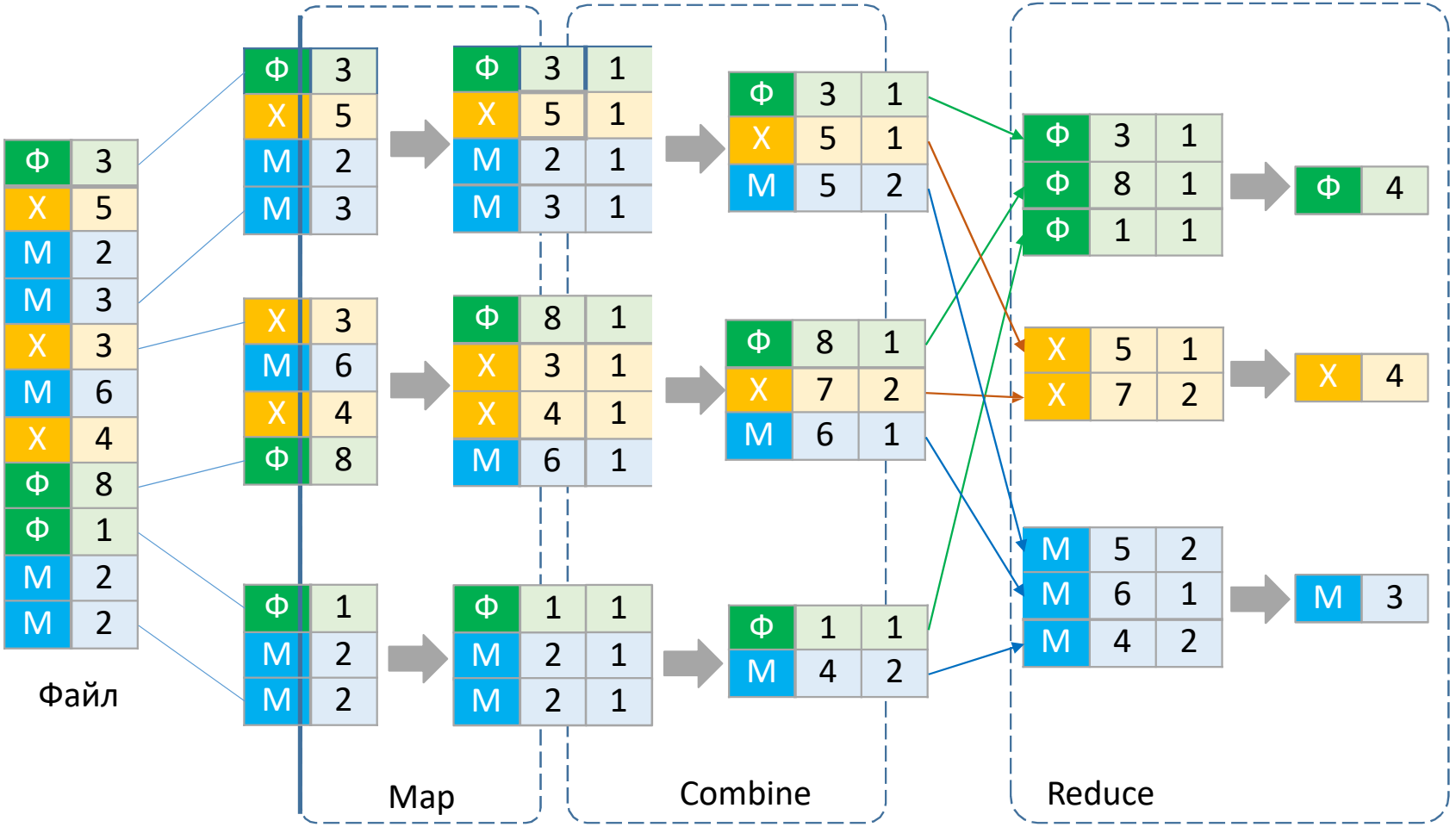
Среднее значение



Среднее значение



Среднее значение



Hadoop MapReduce

Словарь Hadoop MapReduce

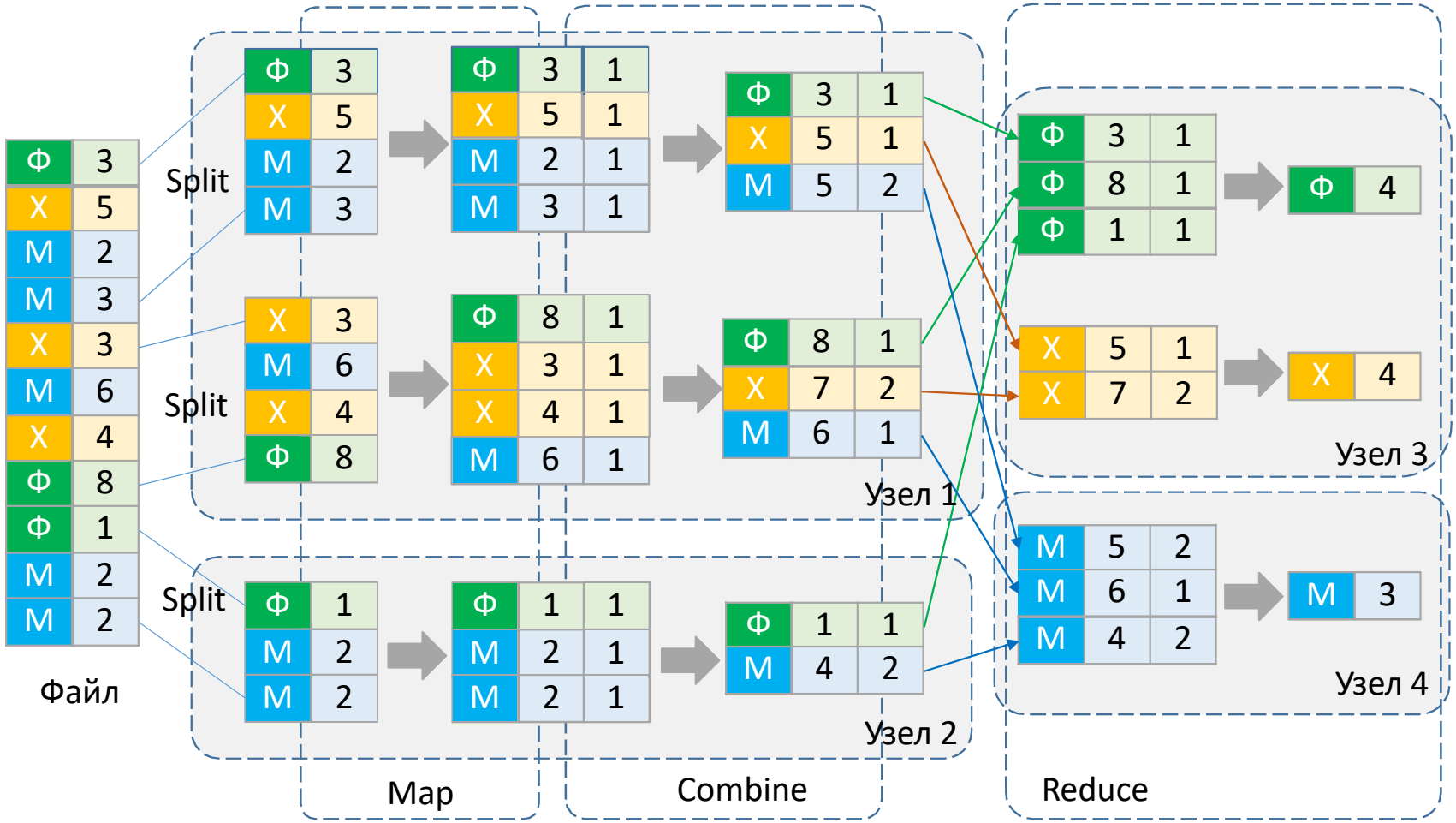
Архитектура/топология

- Split (часть данных)
- Map/combine/reduce задача (task)
- Работа (job)

Программирование

- Mapper
- Combiner
- Partitioner
- Reducer
- InputFormat
- OutputFormat
- Driver

Среднее значение



Особенности MapReduce

- MapReduce работа (job) состоит из стадии map и reduce
- Стадия map – преобразование исходных данных
- Стадия reduce – агрегация данных стадии map
- MapReduce работа (**job**) разбивает исходные данные на независимые логические части – **splits**
- Размер **split** <=> размера блока HDFS. Если больше, то теряется data locality
- Одному **split** соответствует одна **map-**задача
- Задачи запускаются и выполняются параллельно (в идеальном случае)

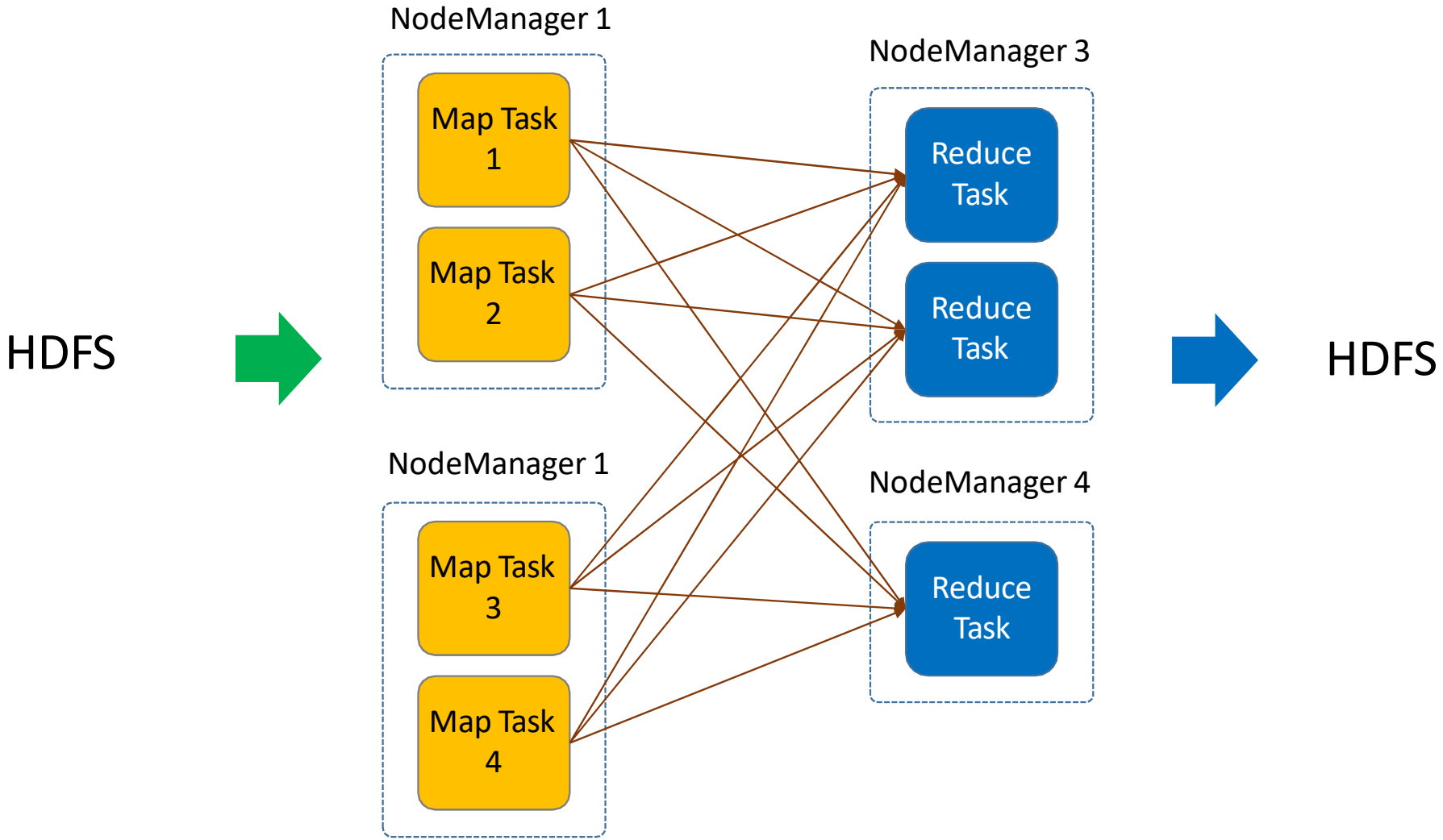
Особенности MapReduce

- Структура входных и выходных данных задач – <ключ, значение>

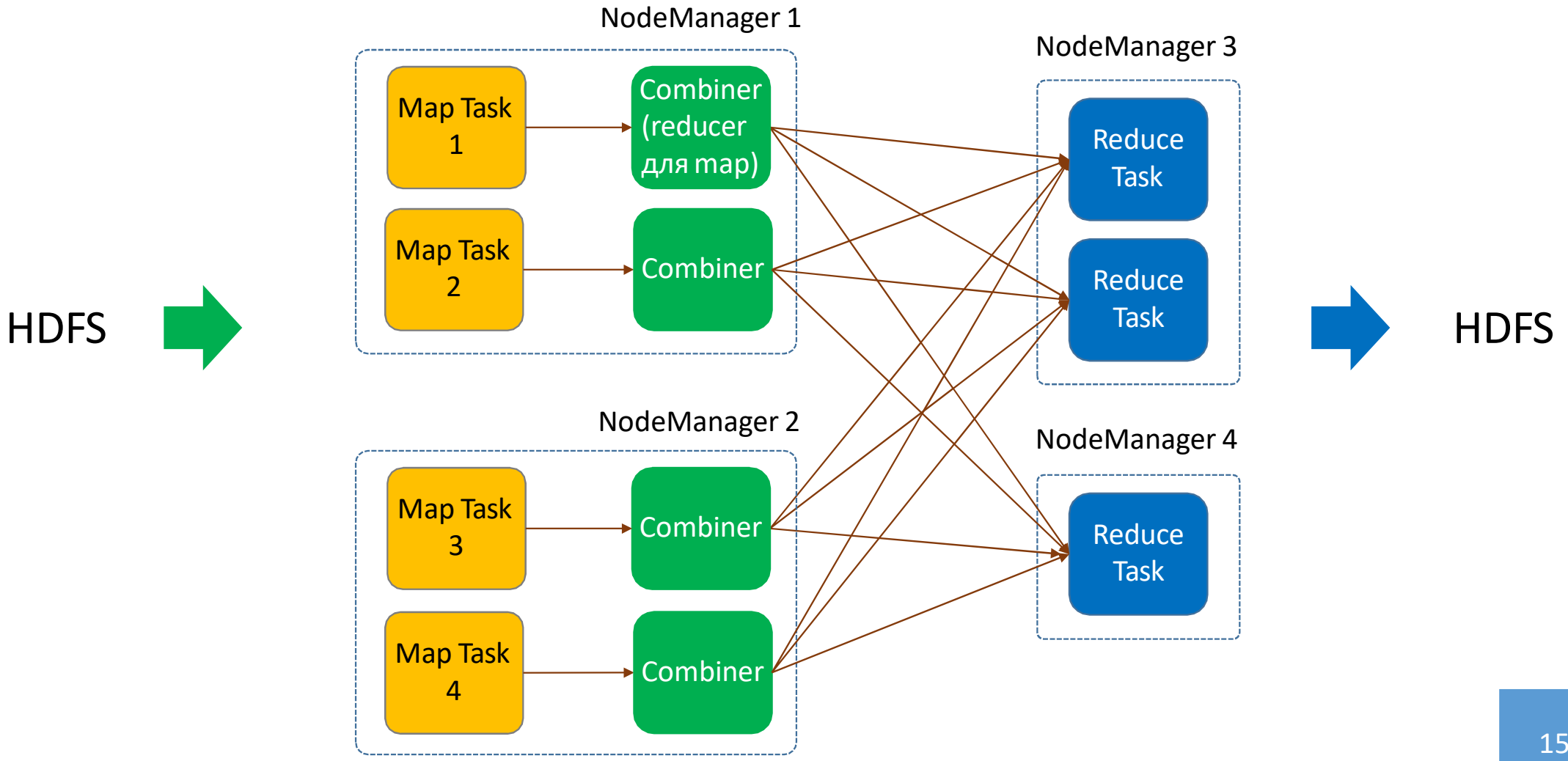
  (input) <k1, v1> -> **map** -> <k2, v2> -> **combine** -> <k2, v2> -> **reduce** -> <k3, v3> (output)

- Задачи **reduce** запускаются при завершении 5% **map**-задач
- Все кортежи с одинаковым ключом находятся на одном **reducer**
- Одна задача – один процесс (JVM)
- Одна задача может иметь несколько потоков, если, например, использовать MultiThreadedMapper
- Каждый **reducer** сохраняет данные в отдельный файл в HDFS (part-0000x)

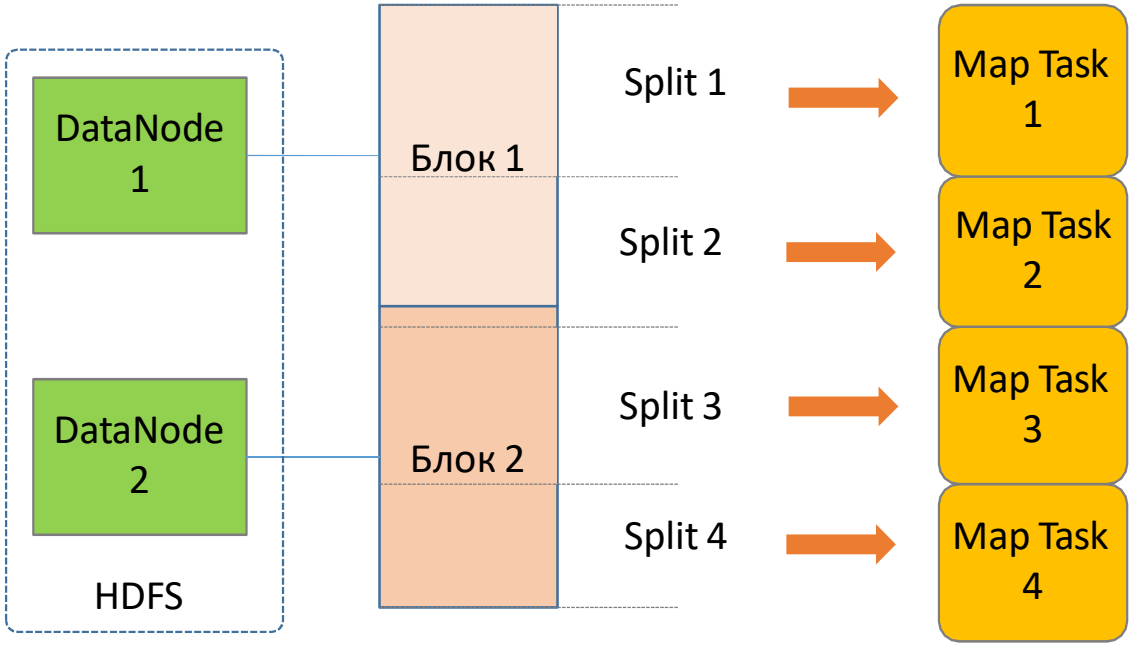
Архитектура MapReduce

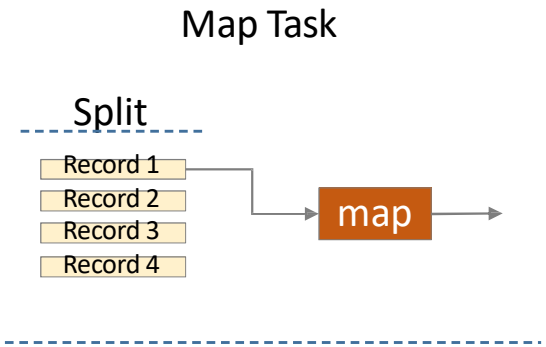


Архитектура MapCombineReduce



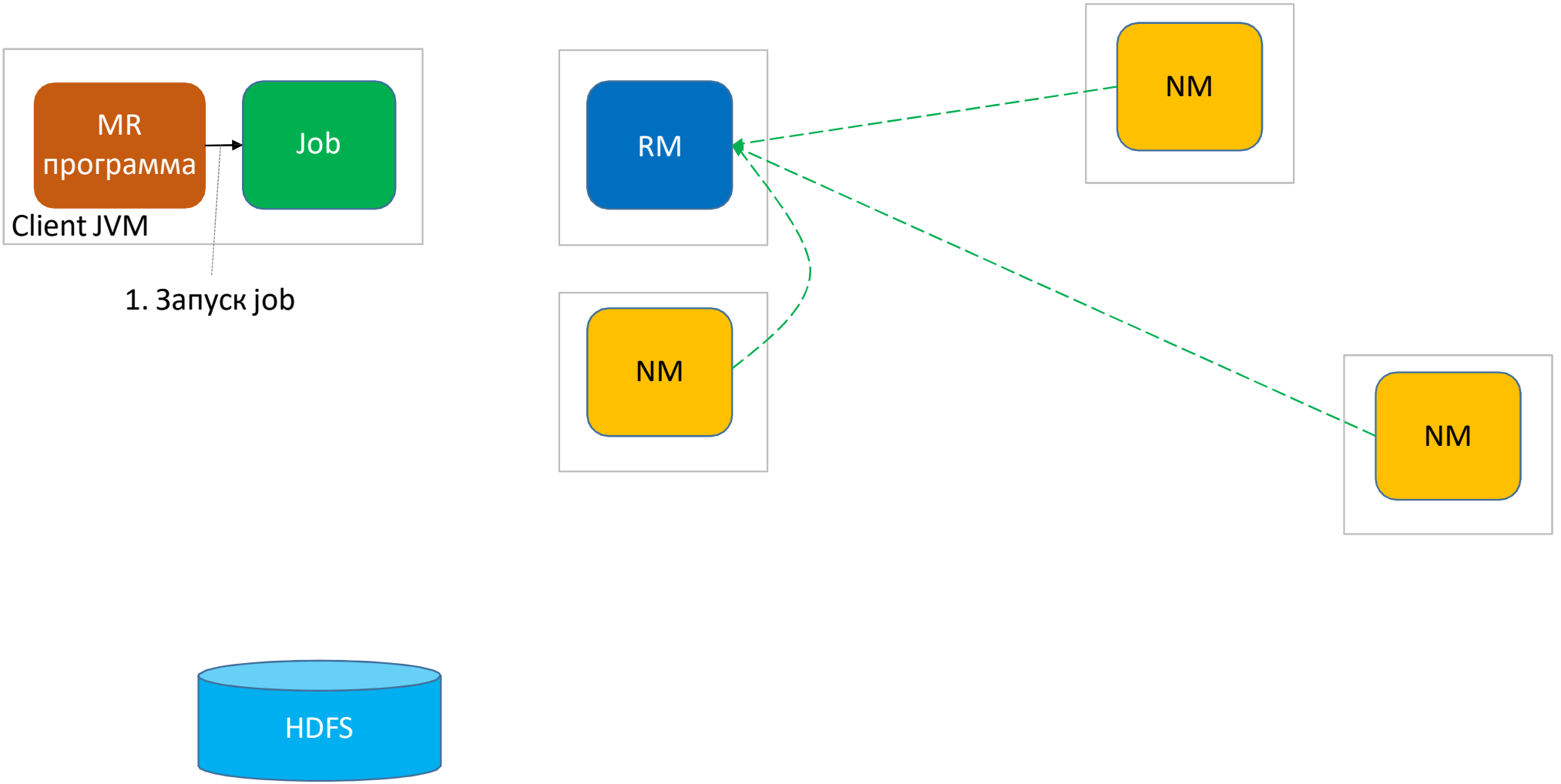
Входные данные для Map Задачи



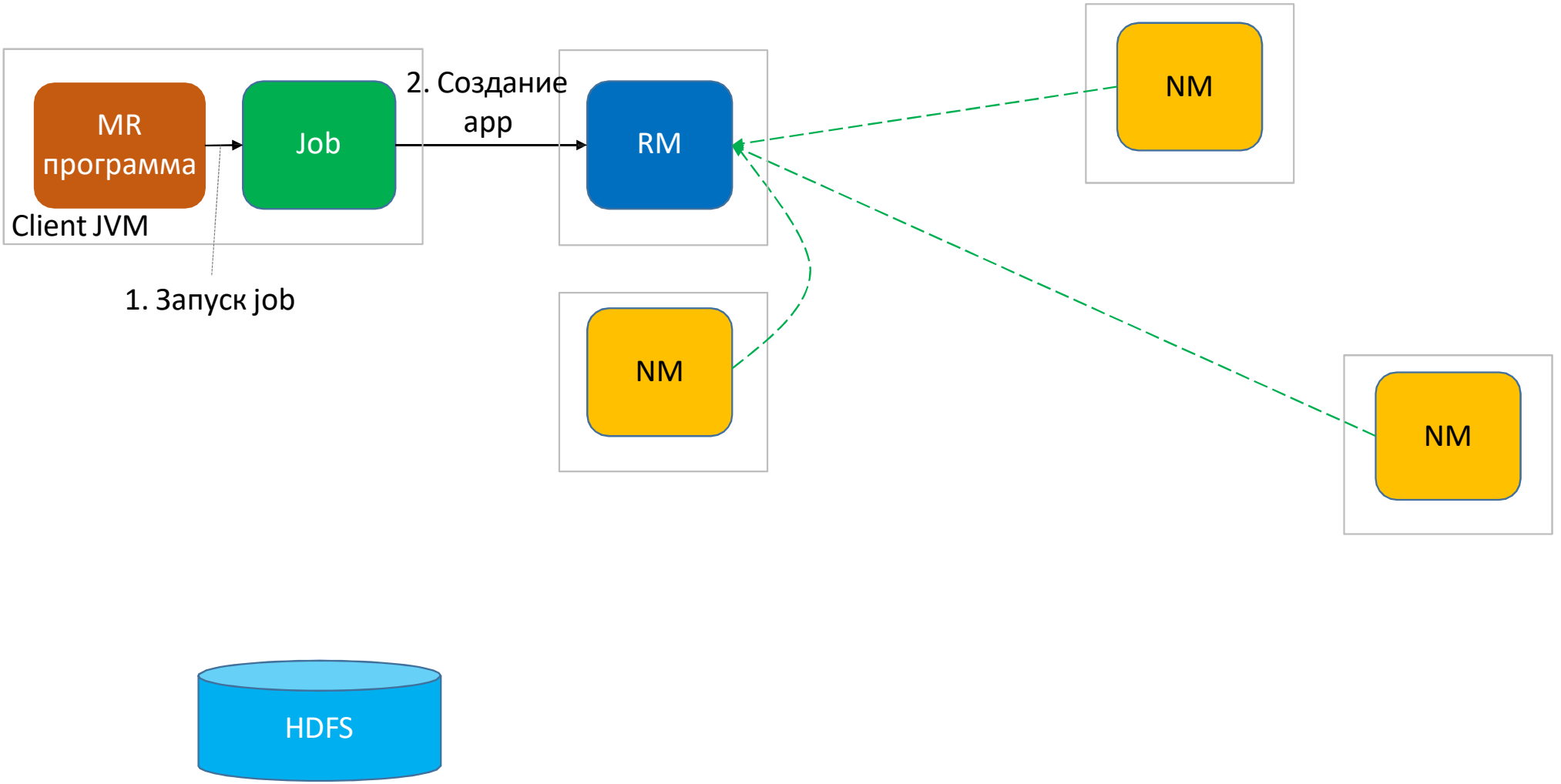


Развертывание приложения MapReduce

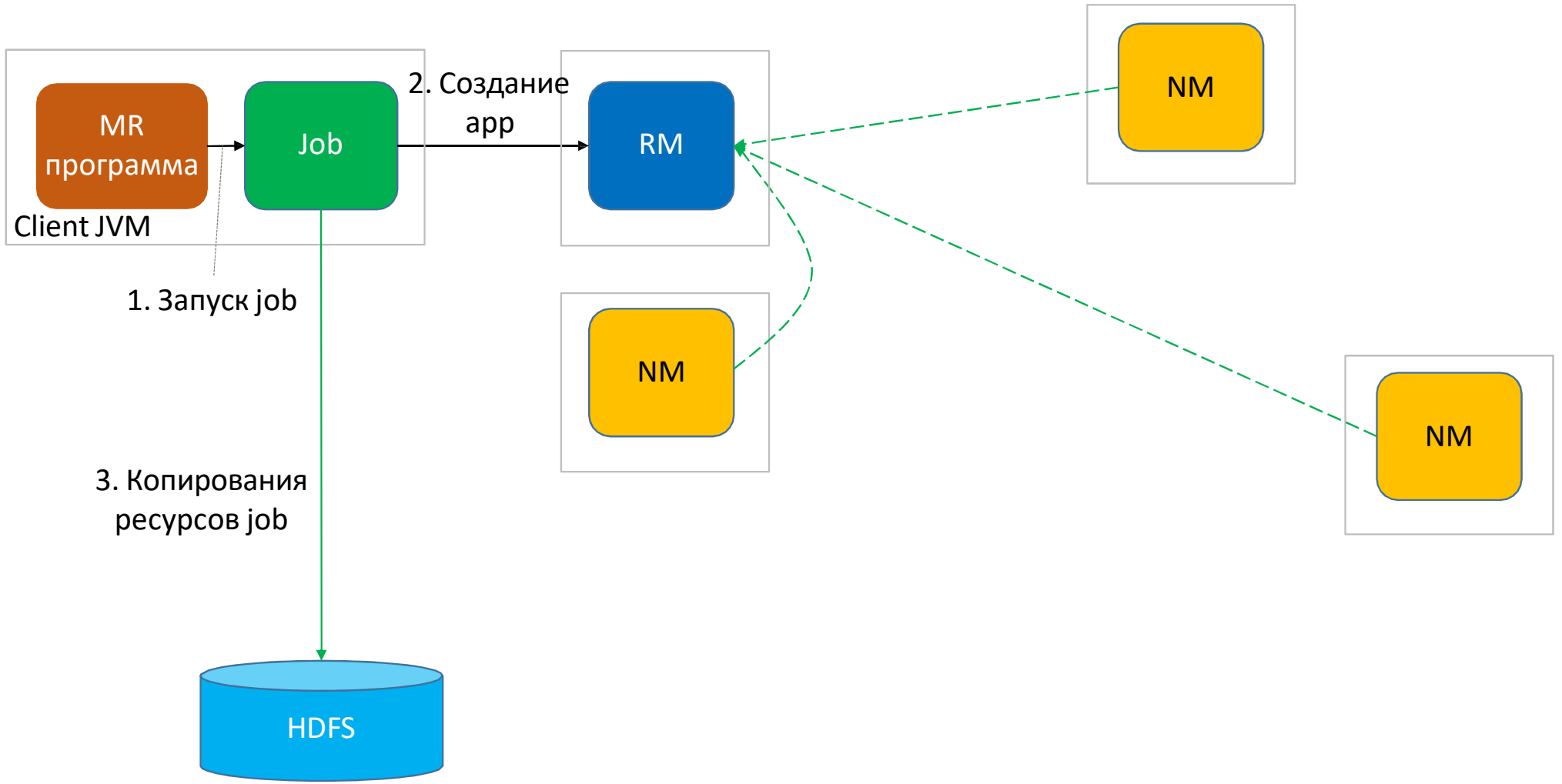
Развертывание приложения MapReduce



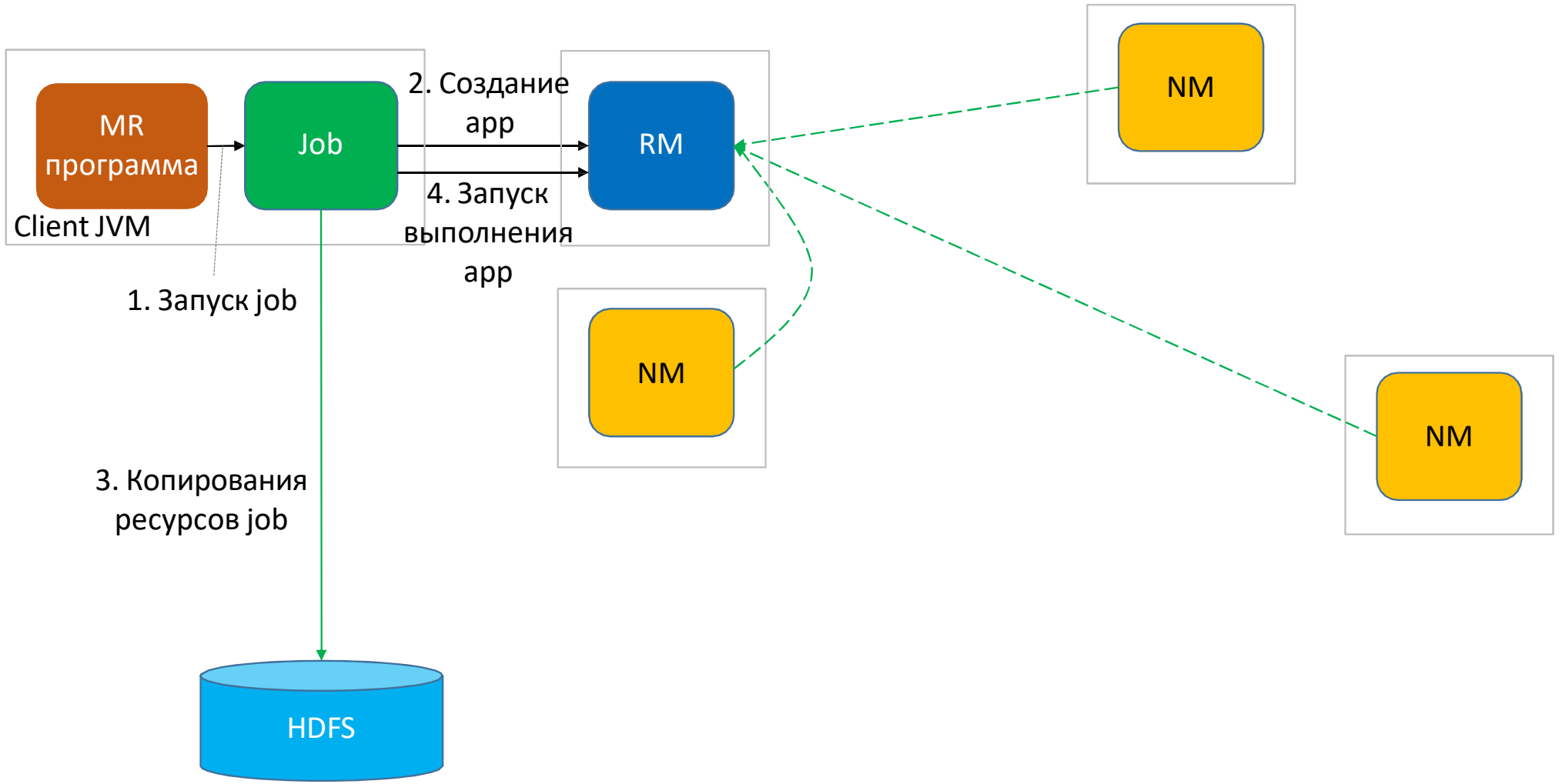
Развертывание приложения MapReduce



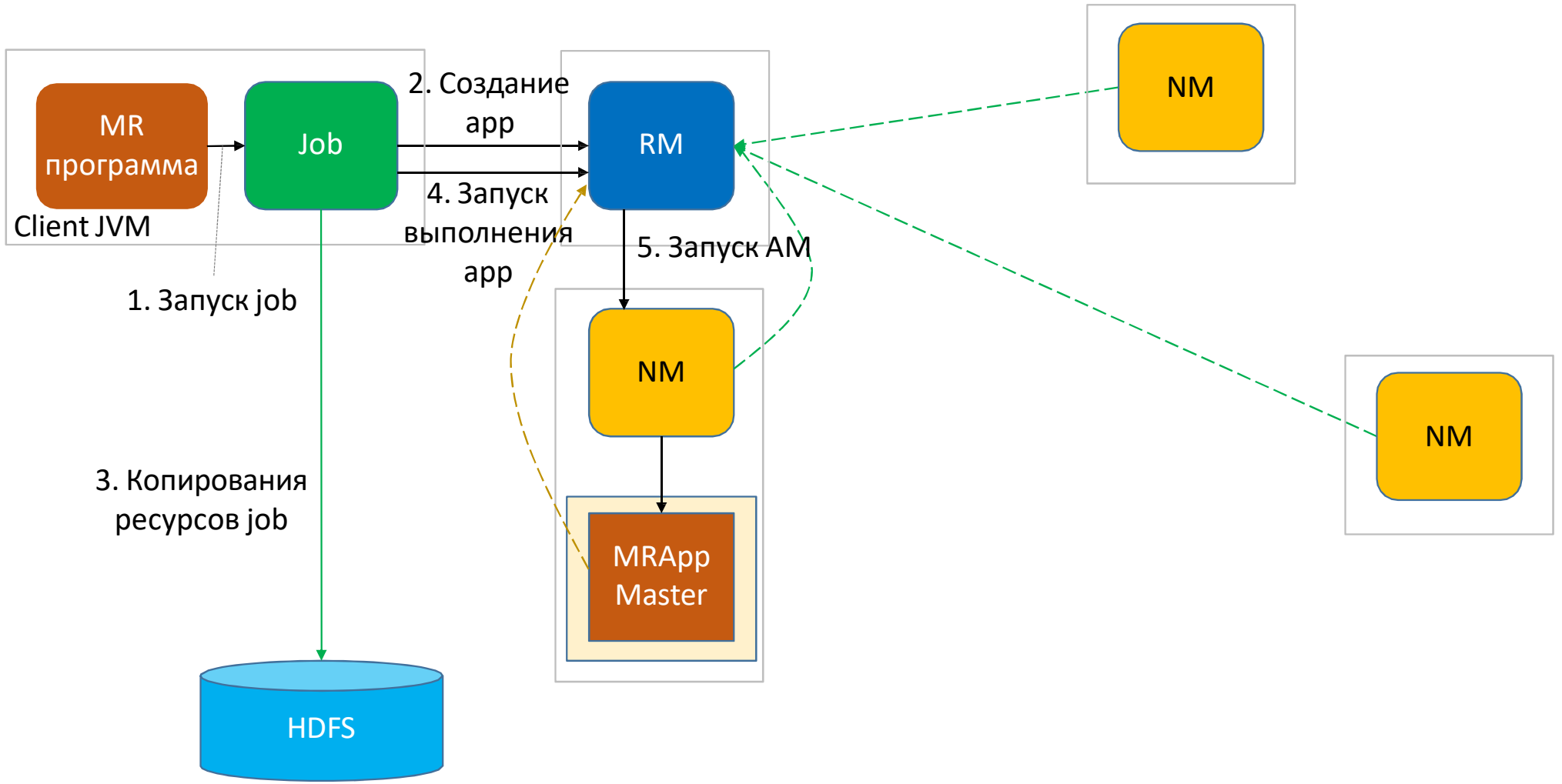
Развертывание приложения MapReduce



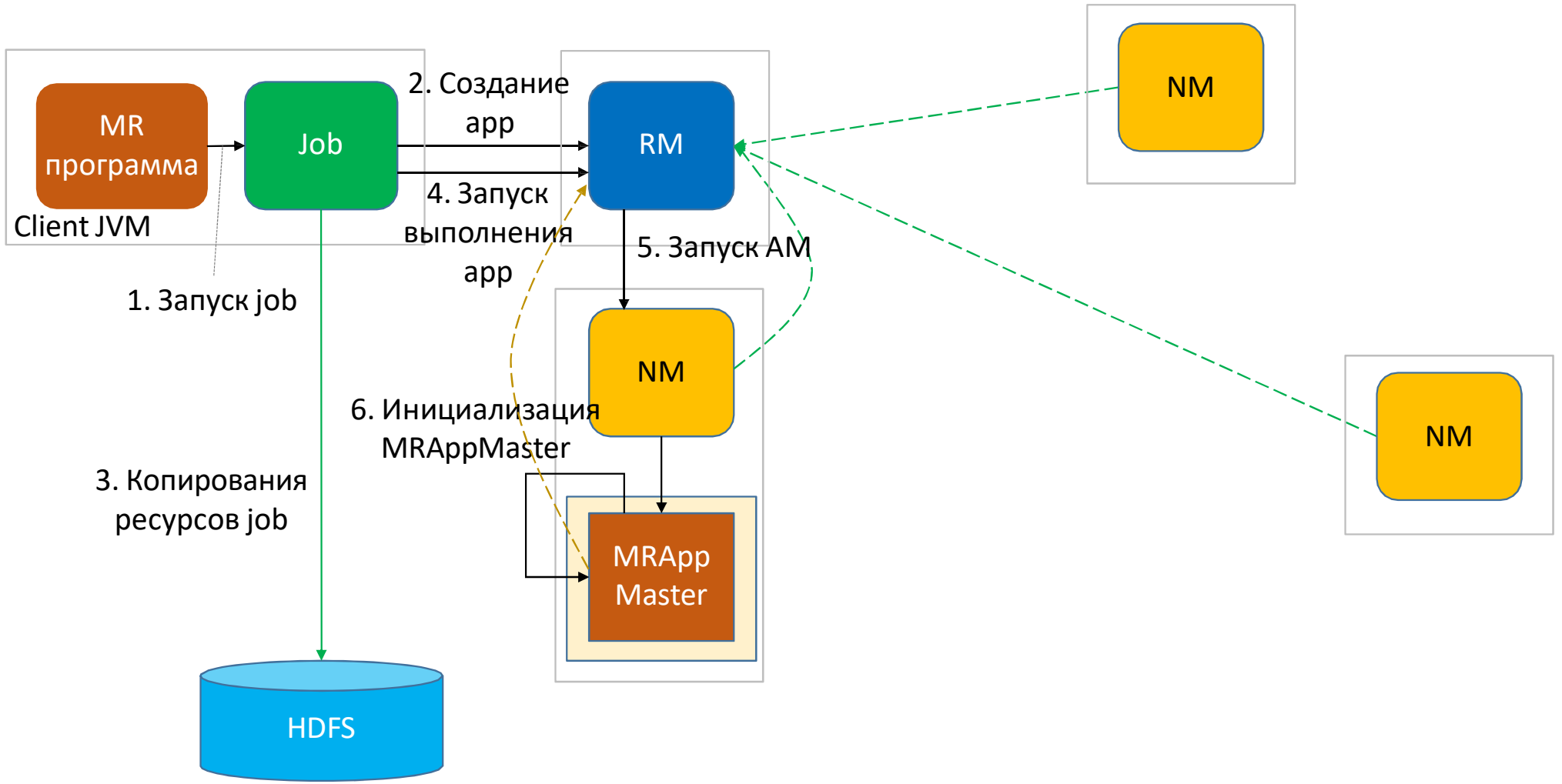
Запуск и выполнение MapReduce



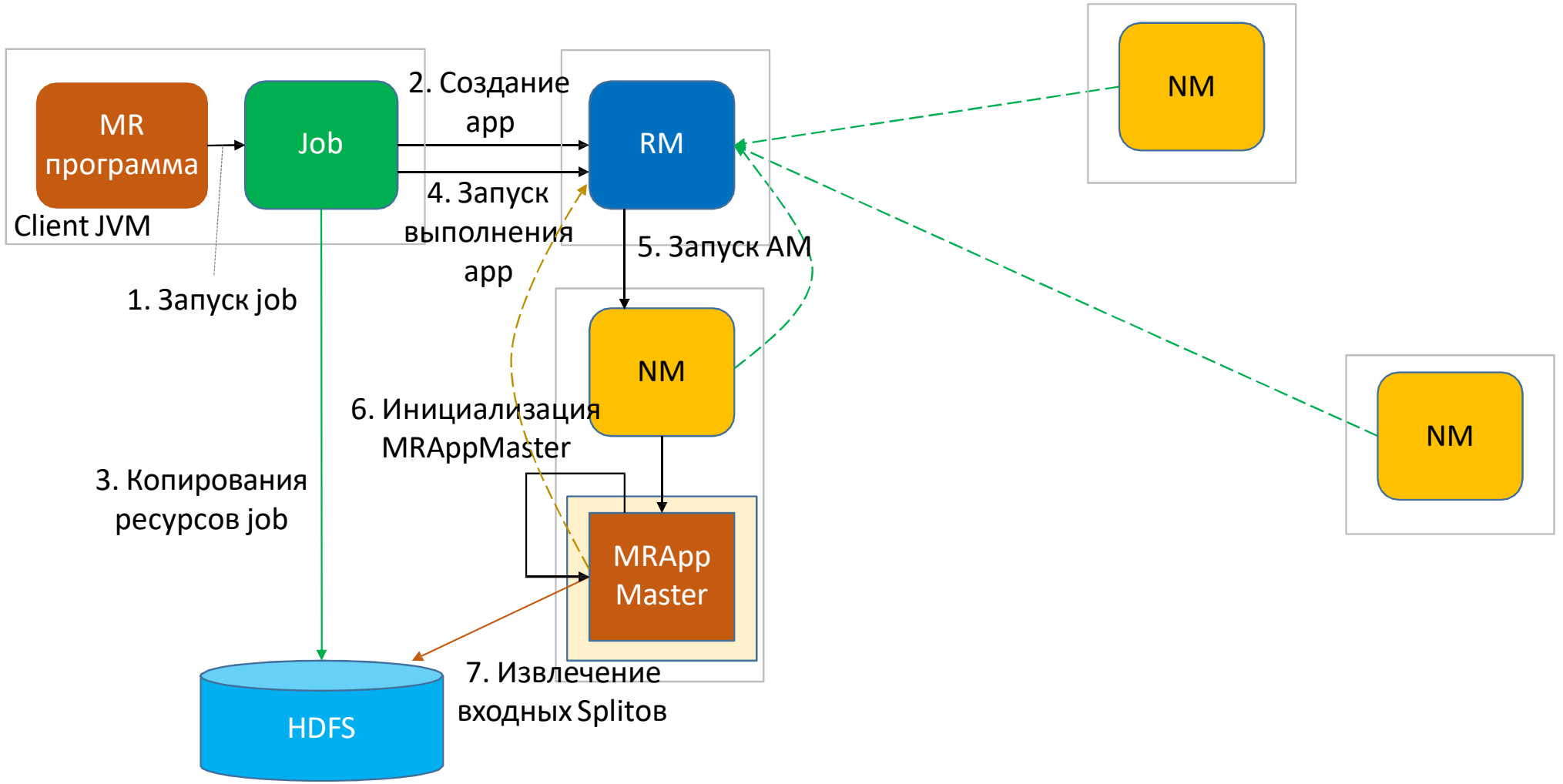
Запуск и выполнение MapReduce



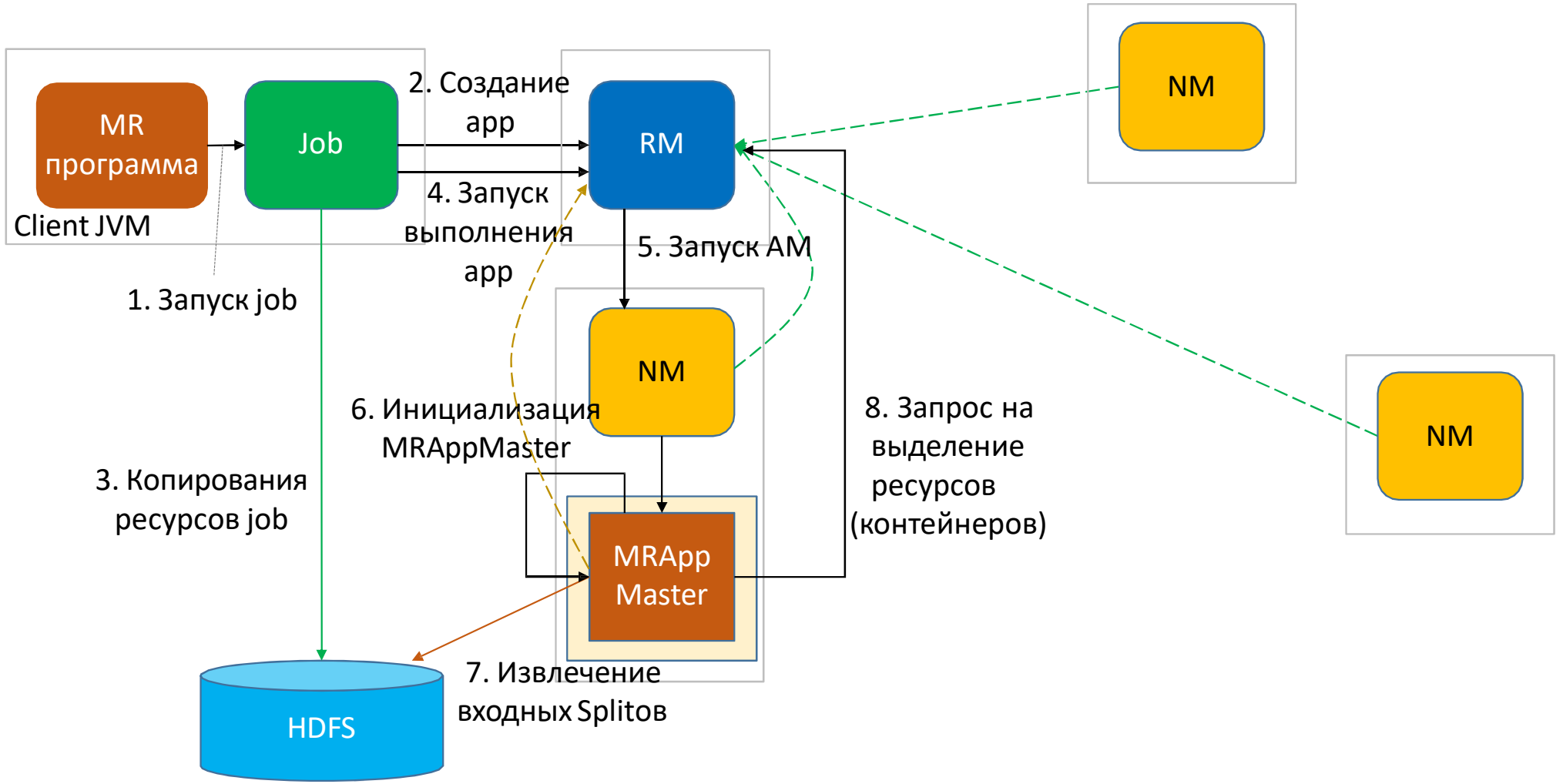
Запуск и выполнение MapReduce



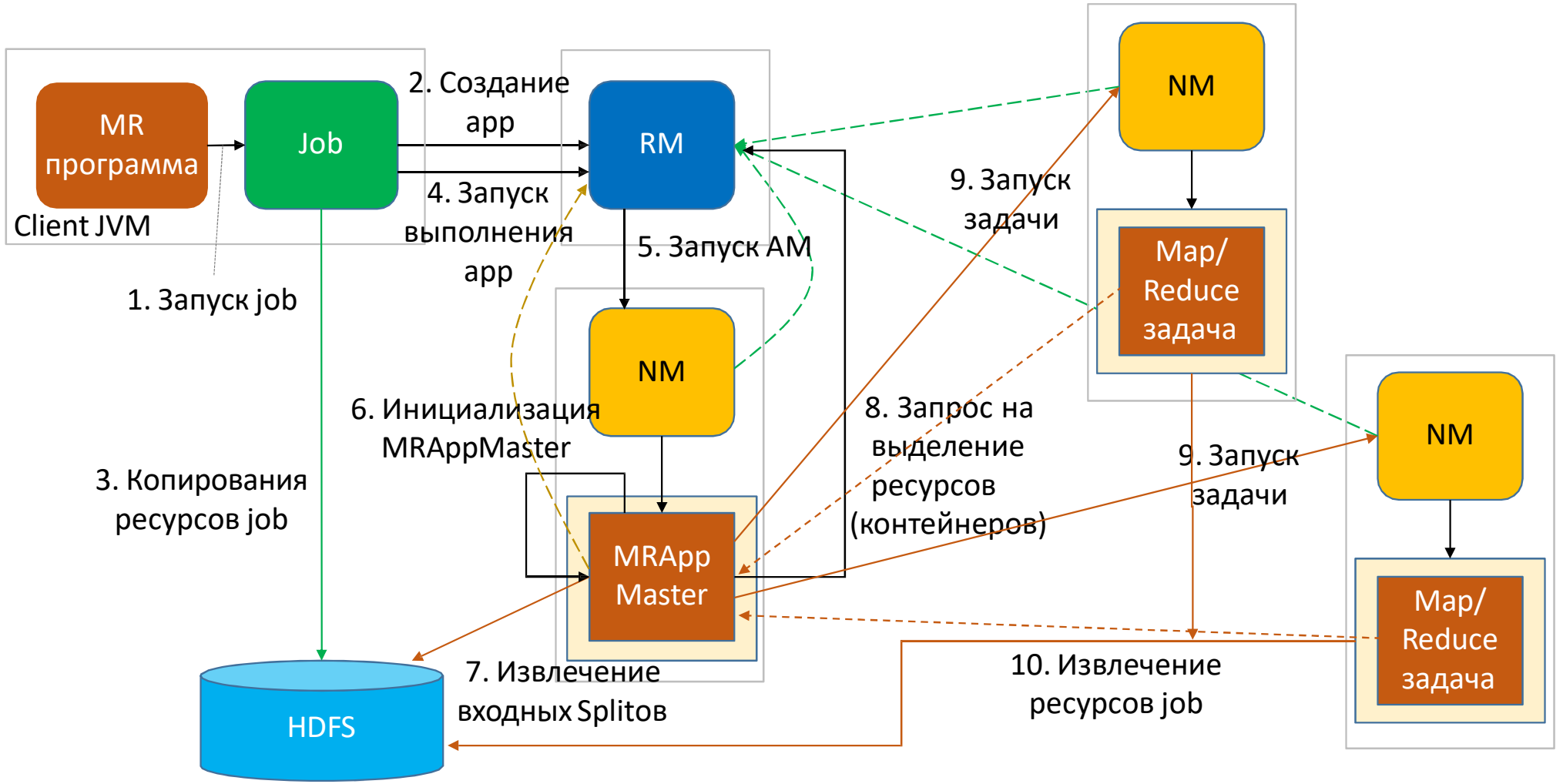
Запуск и выполнение MapReduce



Запуск и выполнение MapReduce

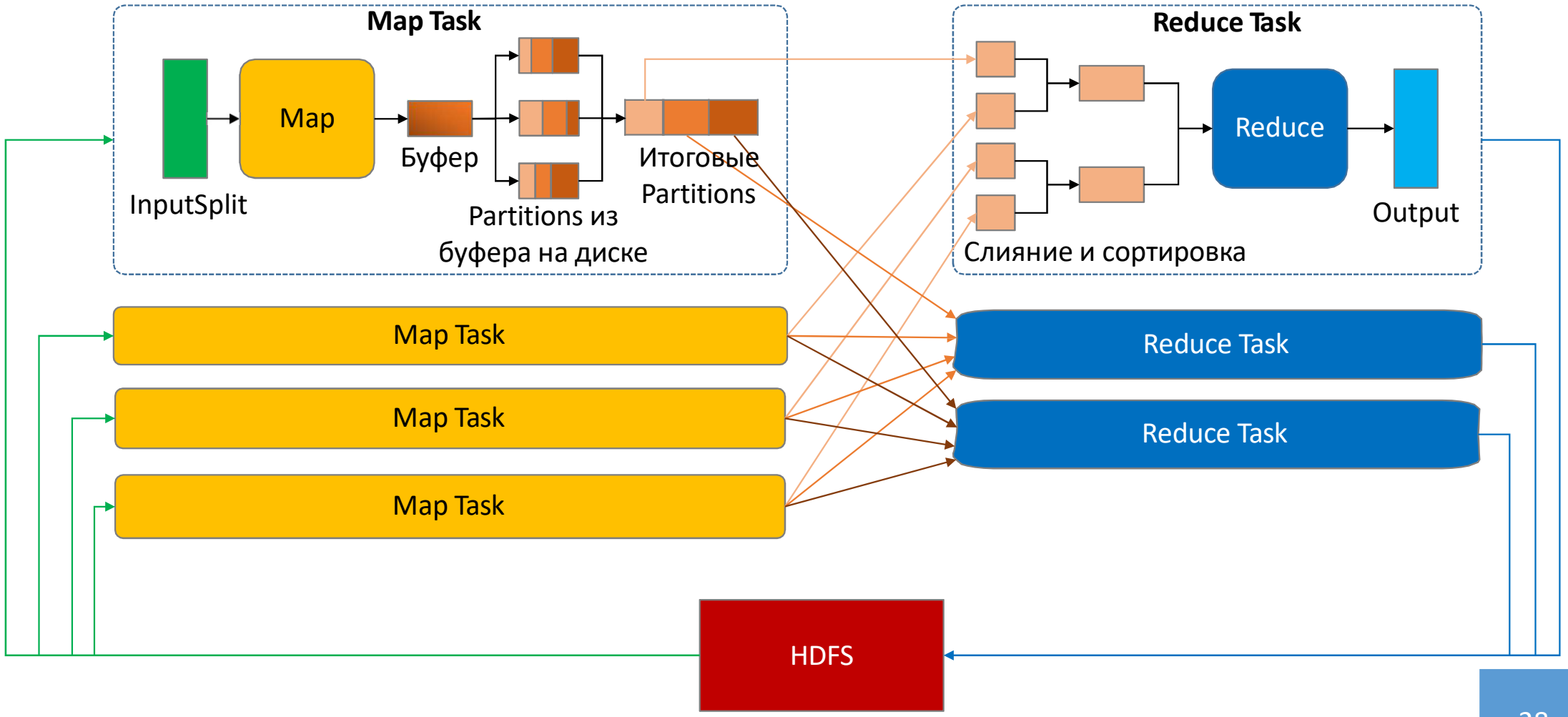


Запуск и выполнение MapReduce

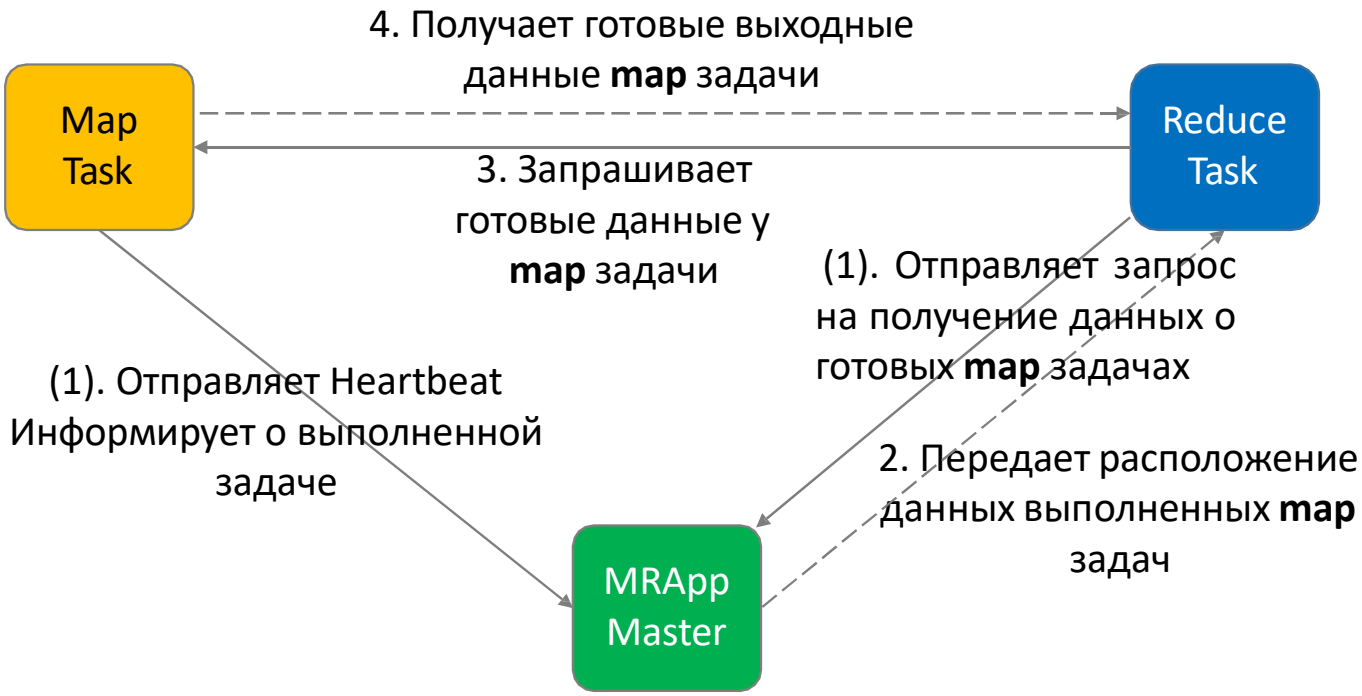


Перетасовка (Shuffling) MapReduce

Перегруппировка (Shuffling) MapReduce

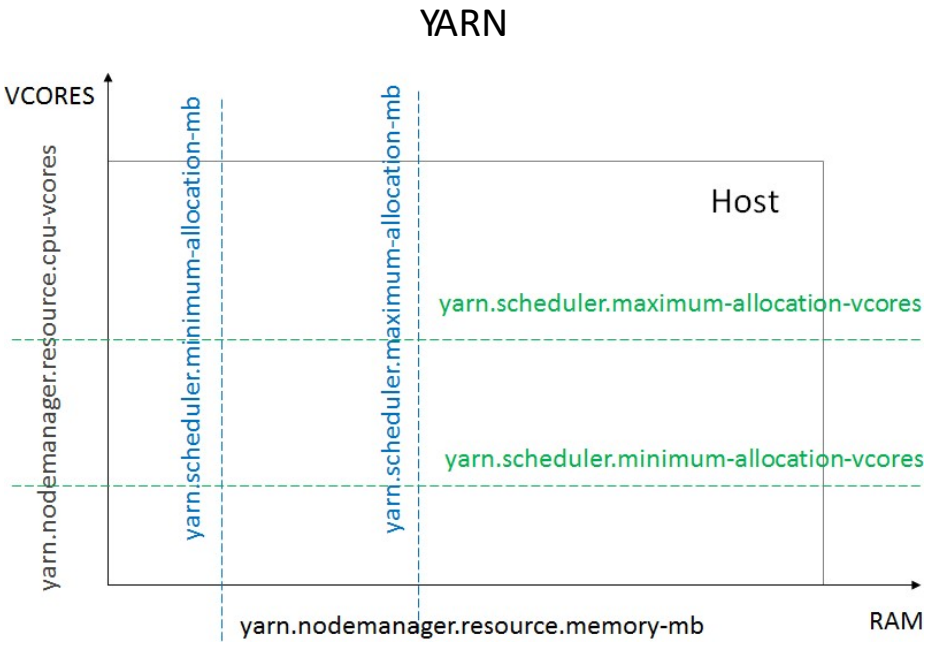


Получение Reduce Task результата Map Task



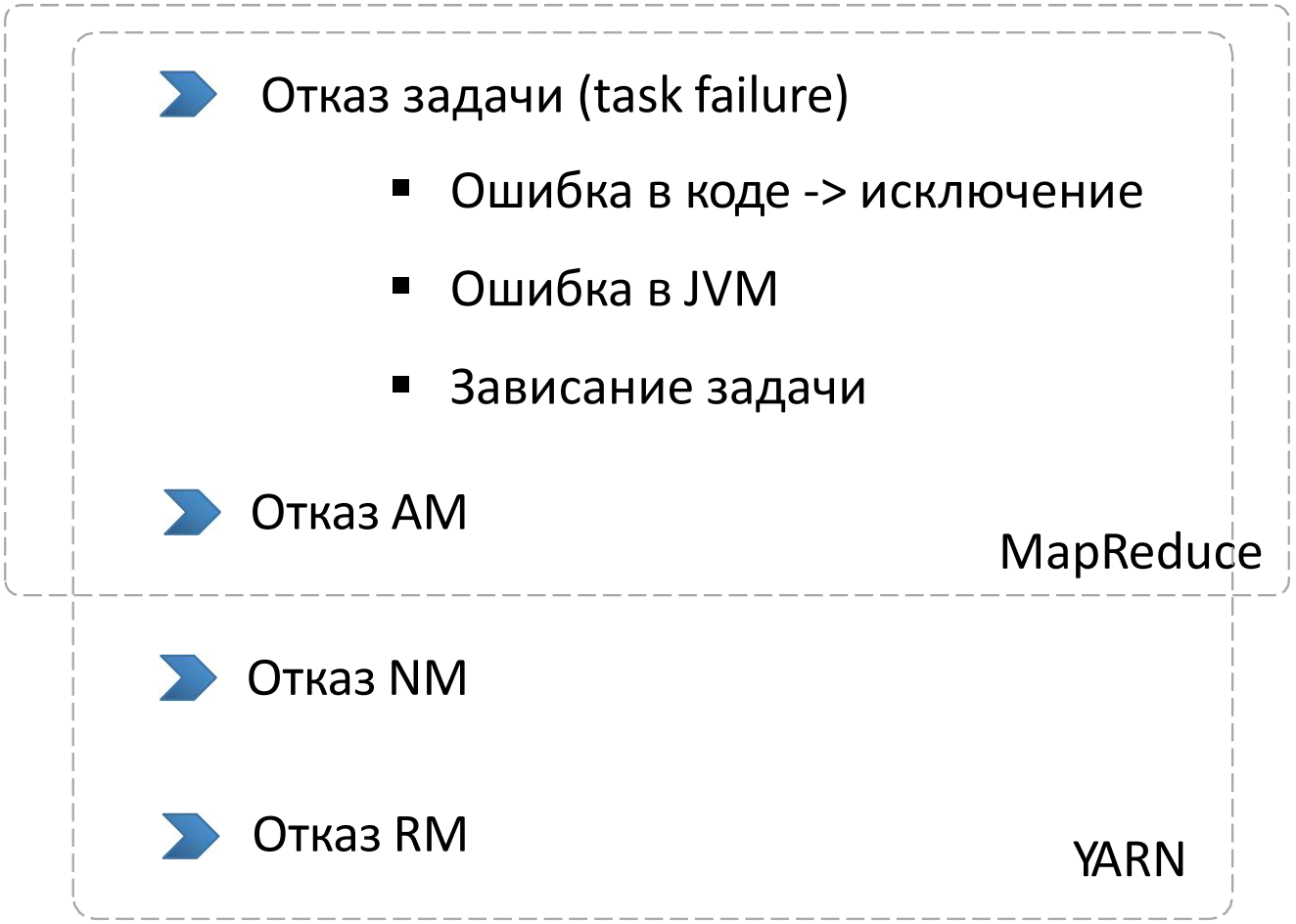
Конфигурирование ресурсов

```
yarn.app.mapreduce.am.resource.cpu-vcores

yarn.app.mapreduce.am.resource.m

mapreduce.map.cpu.vcores

mapreduce.map.memory.mb

mapreduce.reduce.cpu.vcores

mapreduce.reduce.memory.mb

mapreduce.task.io.sort.mb
```



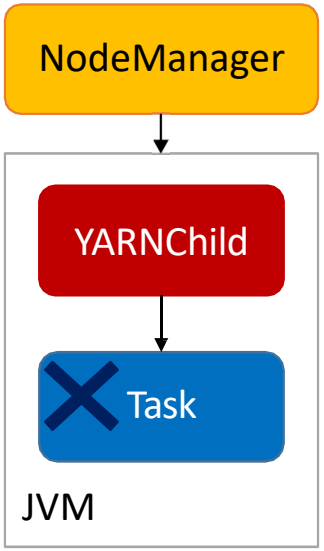
Отказоустойчивость

Выход из строя компонентов MapReduce-YARN



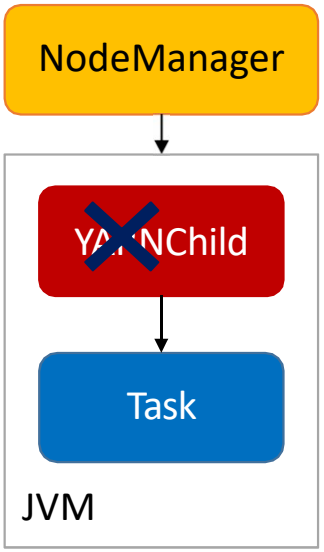
Отказ задачи (task failure). Ошибка в коде -> исключение

- JVM задачи передает информацию AM перед завершением работы
- AM помечает попытку выполнения задачи (task attempt) как failed
- AM освобождает контейнер
- AM пытается повторно выполнить задачу на другом NM



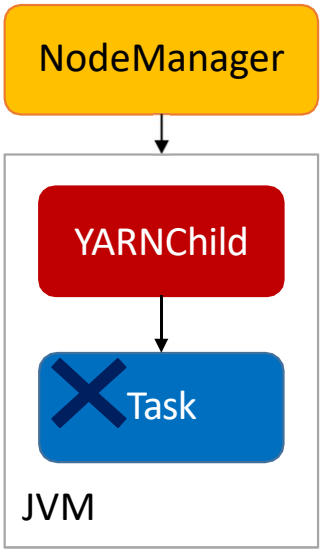
Отказ задачи. Ошибка в JVM

- NN замечает выход JVM
- NN информирует AM
- AM помечает попытку выполнения задачи (task attempt) как failed
- AM пытается повторно выполнить задачу на другом NM



Отказ задачи (task failure). Зависание задачи

- AM замечает, что он не получает обновления от задачи в течение 10 мин
- AM помечает попытку выполнения задачи (task attempt) как failed
- JVM задачи убивается автоматически
- AM пытается повторно выполнить задачу на другом NM

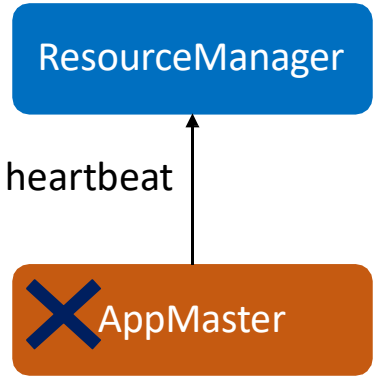


Отказ задачи. Количество попыток

Если 4 раза задача не выполнилась корректно (#attempt=4), то считается, что вся работа (job) потерпела неудачу.

Отказ AM (AM failure)

- RM прекращает получать heartbeat от AM
- RM фиксирует отказ AM
- RM запускает новый экземпляр AM в новом контейнере (запуск через NM)
- Для восстановления состояния выполняемых задач AM использует Job History сервер



#attempts = 2 -> job failed

Отказ NM

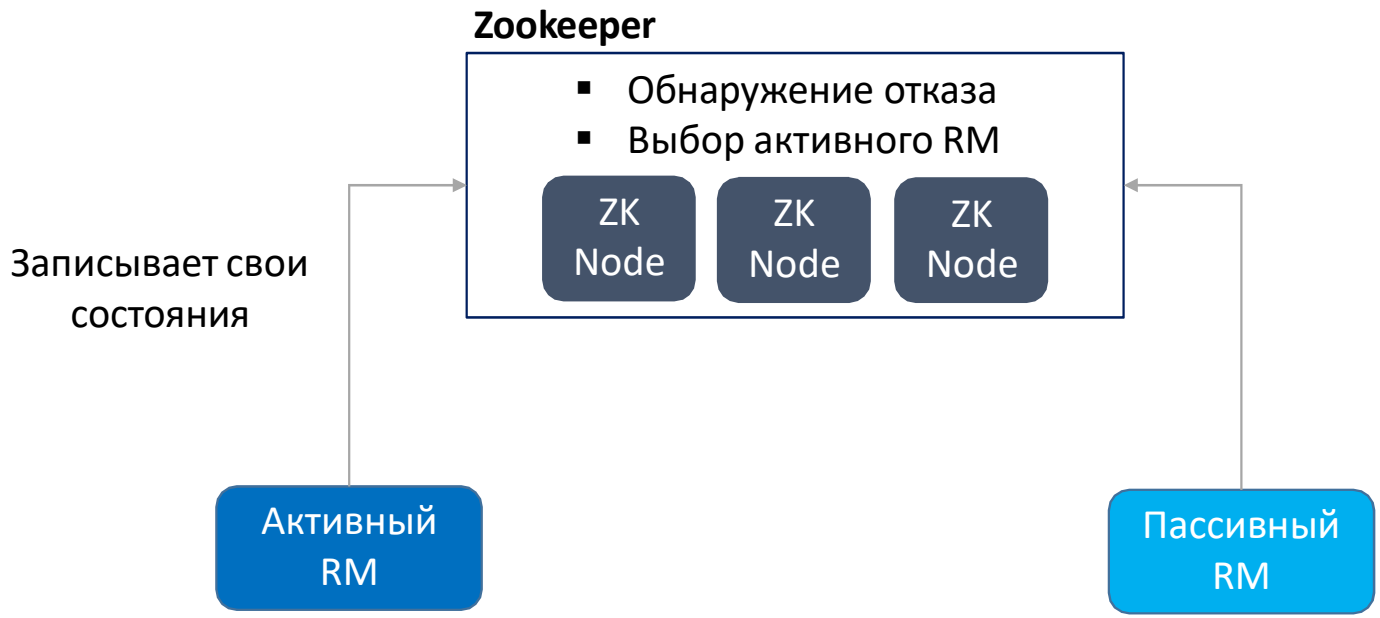
Выход из строя или медленная работа

- RM прекращает получать heartbeat от NM
- RM ждет 10 мин. и не использует этот NM при выделении контейнеров
- Задачи на NM будут восстановлены AM на другом NM

Отказ RM

- RM – единая точка отказа (SPOF)
- Все работы выходят из строя и не могут быть восстановлены
- Для обеспечения отказоустойчивости используется режим High Availability (HA) с активным и пассивным RM

Отказ RM. HA



- ActiveStandbyElector – встроенный в RM детектор отказа (аналог ZKFC в HDFS)
- Приложения могут периодически создавать checkpoint, чтобы избежать потерь при восстановлении
- Клиенты, AMы и NMы пытаются подключиться к RM в Round-Robin манере пока не найдут активный RM (все RMы прописаны в конфигурации на каждом узле)

Отказ RM. HA

Стадии повторного запуска/восстановления RM:

1. Восстановление состояния RM

   Загружает информацию из ZK о приложениях/попытках выполнения

   Повторно запускает ранее незавершенные приложения (Hadoop 2.4.0)

2. Восстановление запущенных работ (Hadoop 2.6.0)

   RM собирает информацию о статусах контейнеров с NM и запрошенных контейнерах от AM. Предыдущие работающие приложения не теряют проделанную работу

RM не запускает приложение повторно, если приложение отмечено статусом завершения: failed, killed или finished

Отказ RM. HA

ZK хранит следующую информацию:

- Метаданные контекста приложения (формируемые при запуске приложения клиентом)
- Финальные статусы приложения (failed, killed, finished) и данные диагностики после завершения приложения
- Ключи безопасности, токены для работы в защищенной среде

Вместо ZK для хранения может быть использована локальная FS, HDFS или LevelDB

Отказ RM. HA. Стадия 1

- Клиенты и NM пытаются достучаться до RM, периодически отравляя запросы
- После запуска RM и загрузки данных из ZK он отправляет re-sync команду всем NMам и AMам (через их heartbeatы)
- Получив re-sync NM уничтожает управляемые им контейнеры и повторно регистрируются на RM
- AM также завершается при получении re-sync
- Создается новая попытка запуска AM для всех приложений, которые не были завершены

Отказ RM. HA. Стадия 2

- RM восстанавливает свое рабочее состояние, получая информацию о статусах контейнеров от всех NMов
- NM не уничтожает контейнеры при re-sync команде
- NM повторно регистрируется на RM и отправляет ему статусы контейнеров
- RM не уничтожает AM
- AM синхронизируется с RM и продолжает свою работу

Источники

Hadoop: The Definitive Guide, 4th Edition (book)

Apache Hadoop YARN (doc)

MapReduce Tutorial (doc)

ResourceManager High Availability (doc)

ResourceManager Restart (doc)

Hadoop (github source code)