ТЕХНОСФЕРА

Лекция 14

YARN
Hadoop 2.0

Евгений Чернов

YARN

1. Компоненты ***YARN***
2. Детали про выполнение задач ***MapReduce*** на **YARN**
    1. Job Submission
    2. Job Initialization
    3. Tasks Assignment
    4. Tasks' Memory
    5. Status Updates
    6. Failure Recovery

YARN

- **Y**et **A**nother **R**esource **N**egotiator
- Разработан в Yahoo! в 2010 году
- Берет на себя функции управления ресурсами
- Может запускать различные типы задач (MapReduce один из вариантов)

Недостатки MapReduce v1

- Жесткое разделение ресурсов кластера
- Нет возможности разделять ресурсы с не MapReduce задачами
- Ограничения в маштабируемости:
  - 4 000 серверов
  - 40 000 запущенных задач

Жесткое разделение ресурсов

- Пусть у нас есть 10 серверов по 16 ядер
- 1 ядро для TaskTracker
- Всего 150 ядер
- Пусть ресурсы делятся следующим образом:
  - 100 мапперов
  - 50 редьюсеров

Жесткое разделение ресурсов

Пусть есть задача:

- нужно 300 мапперов
- каждый маппер работает 10 минут
- если все данные для редьюсеров обрабатывать одним редьюсером, то это займет 100 минут

Какое время потребуется для работы задачи?

$$\frac{300}{100} * 10 + \frac{100}{50} = 32 \text{ мин}$$

- 30 минут 50 редьюсеров простаивали
- 2 минуты 100 мапперов простаивали

Жесткое разделение ресурсов

Меняем распределение мапперов и редьюсеров:

- мапперов: 120
- редьюсеров: 30

Время работы задачи:

$$\frac{300}{120} * 10 + \frac{100}{30} = 28{,}5 \text{ мин}$$

Еще меняем:

- мапперов: 140
- редьюсеров: 10

Время работы задачи:

$$\frac{300}{140} * 10 + \frac{100}{10} = 21{,}5 + 10 = 31{,}5 \text{ мин}$$

Жесткое разделение ресурсов

Можно найти оптимальное разбиение для данной задачи, найдя минимум функции:

$$\begin{cases} T(x,y) = \dfrac{300}{x} * 10 + \dfrac{100}{y} \\ x + y = 150 \end{cases}$$

Но! Разбиение будет оптимально для данной конкретной задачи. Для другой задачи оно может быть **совсем** неоптимальным.

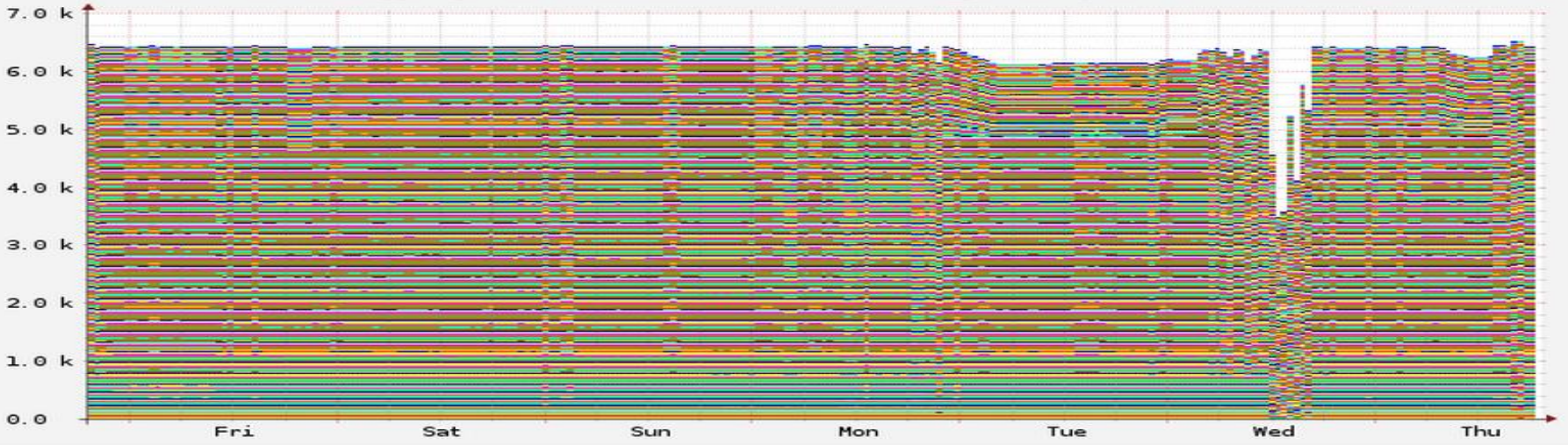
Динамическое распределение ресурсов

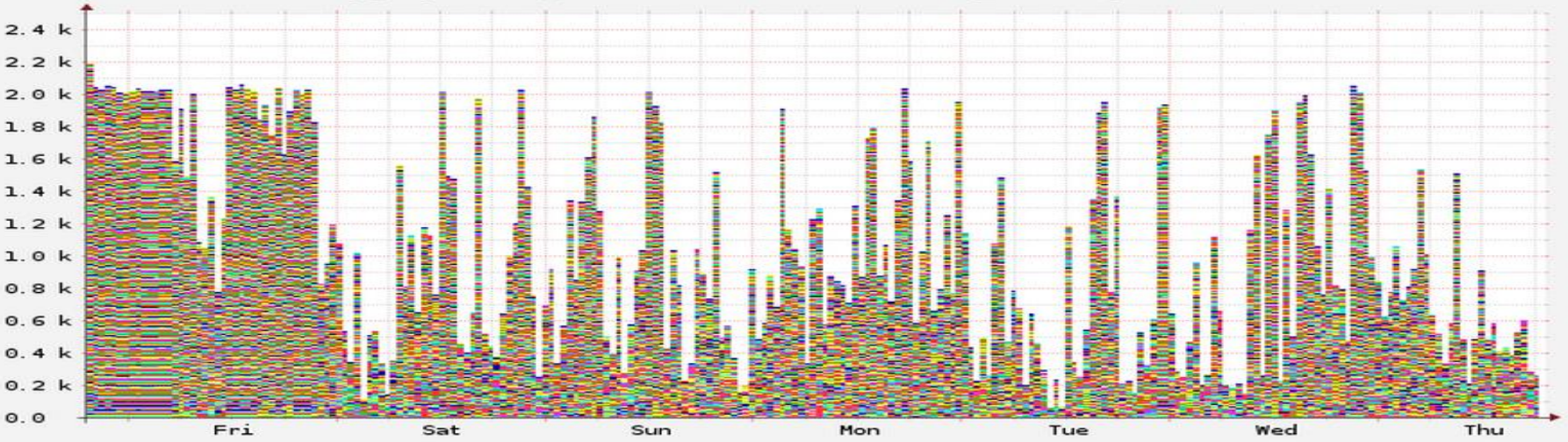
Пусть любой ресурс может быть как маппером, так и редьюсером

Время работы задачи:

$$\frac{300}{150} * 10 + \frac{100}{150} = 20{,}7 \text{ мин}$$

Проблема разделения ресурсов





YARN

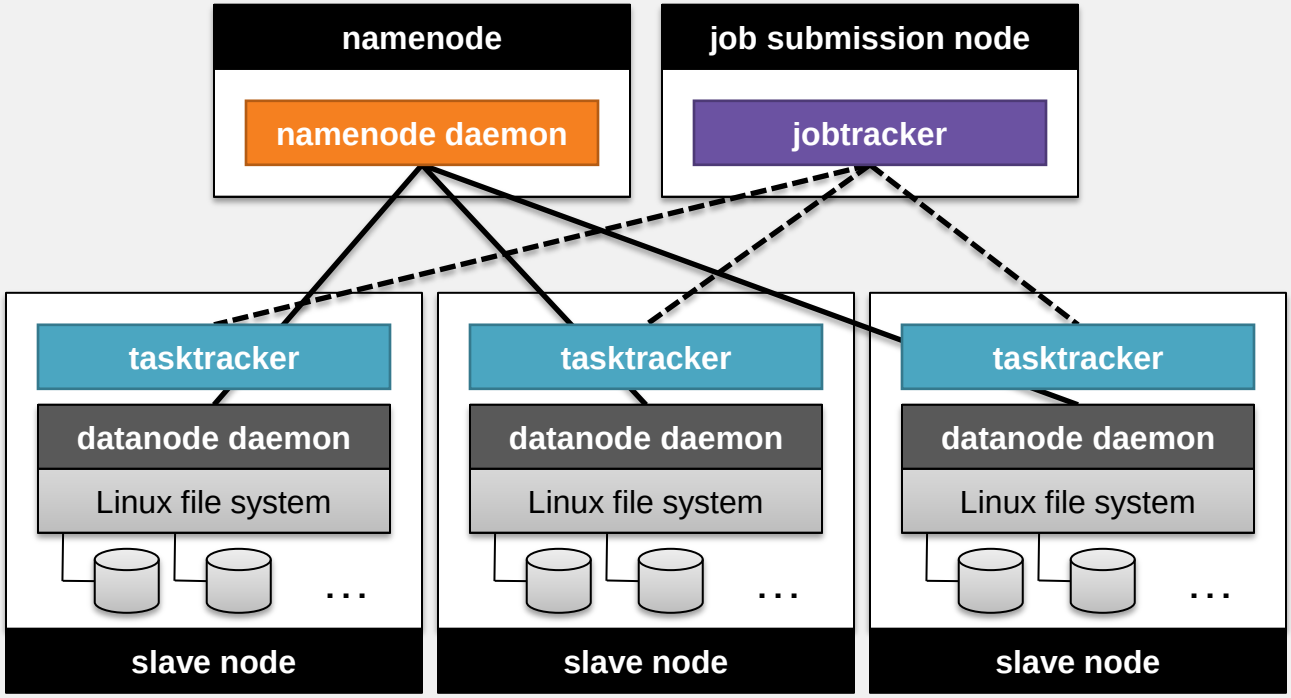
Нет понятия “слоты”

- Сервер имеет “ресурсы” (память, процессор)

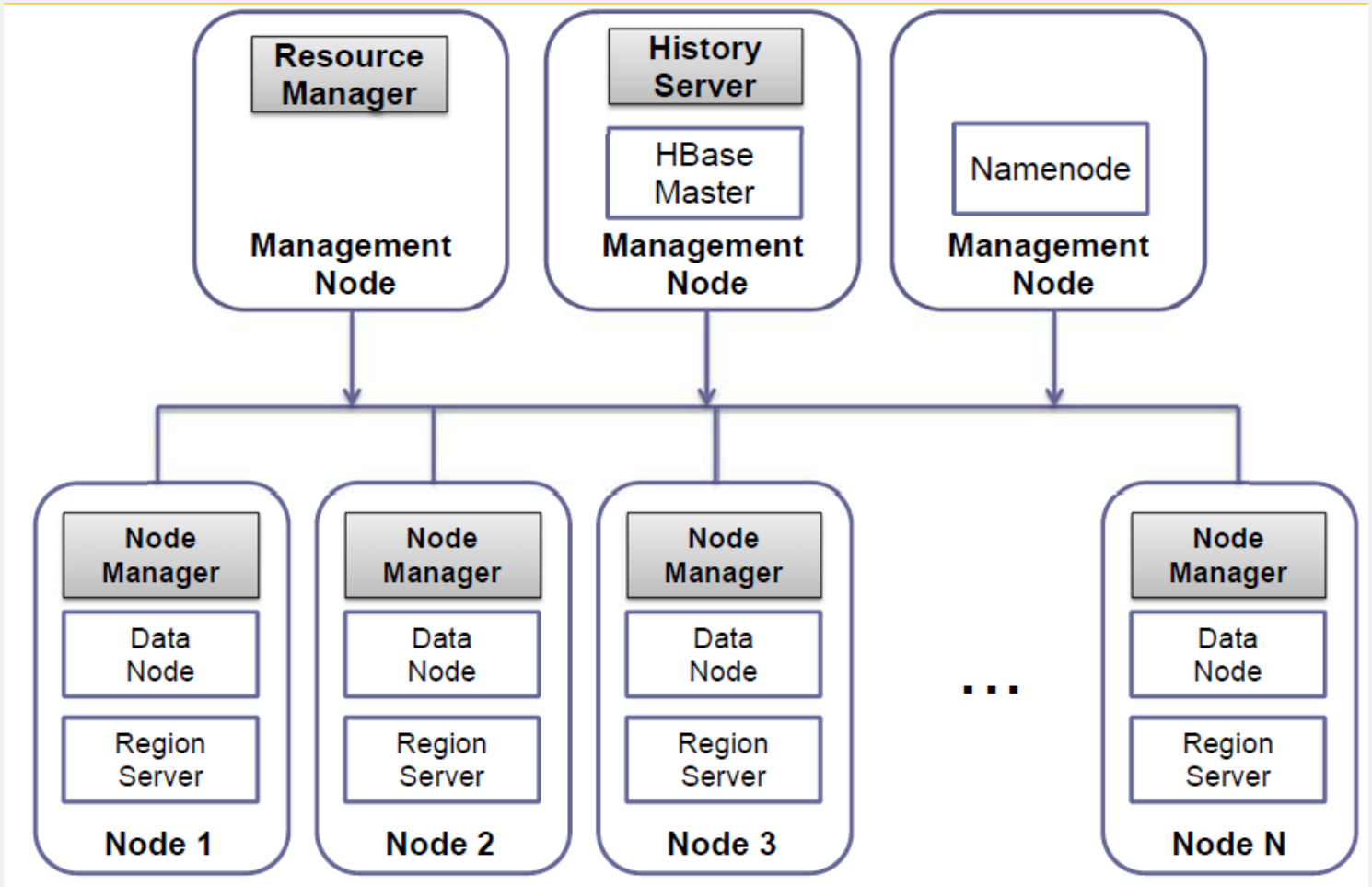
Нет деления на мапперы и редьюсеры

- Можно запускать не только MR задачи

JobTracker и TaskTracker



Пример схемы демонов YARN



Компоненты YARN

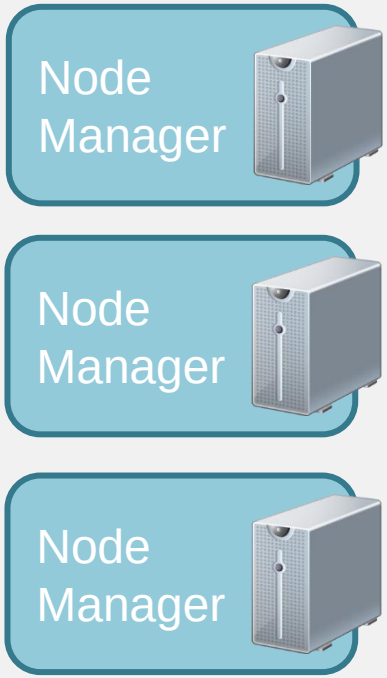
Resource Manager (RM)

- Запущен на отдельном сервере
- Управляет глобальным распределением ресурсов
- Разрешает конфликты между конкурирующими приложениями

Компоненты YARN

Node Manager (NM)

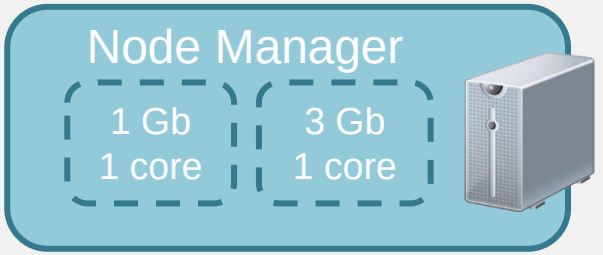
- Запущен на всех нодах кластера
- Взаимодействует с RM



Компоненты YARN

Containers

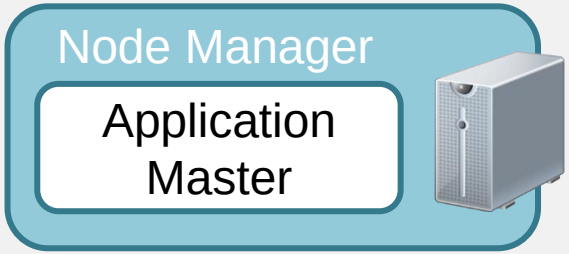
- Создается RM по запросу
- Захватывает определенное число ресурсов на ноде (память, CPU)
- Приложение запускается на одном или нескольких containers

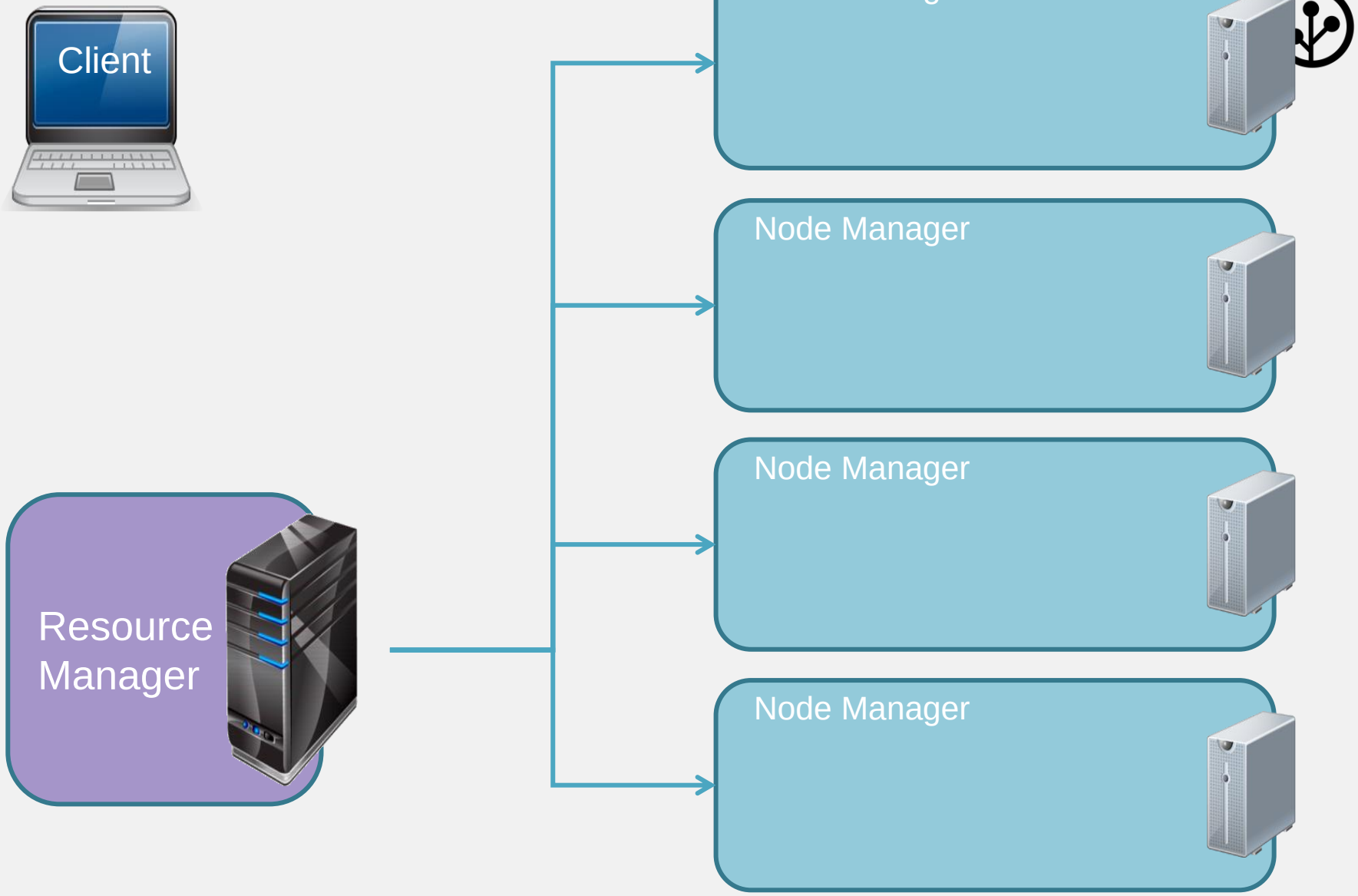


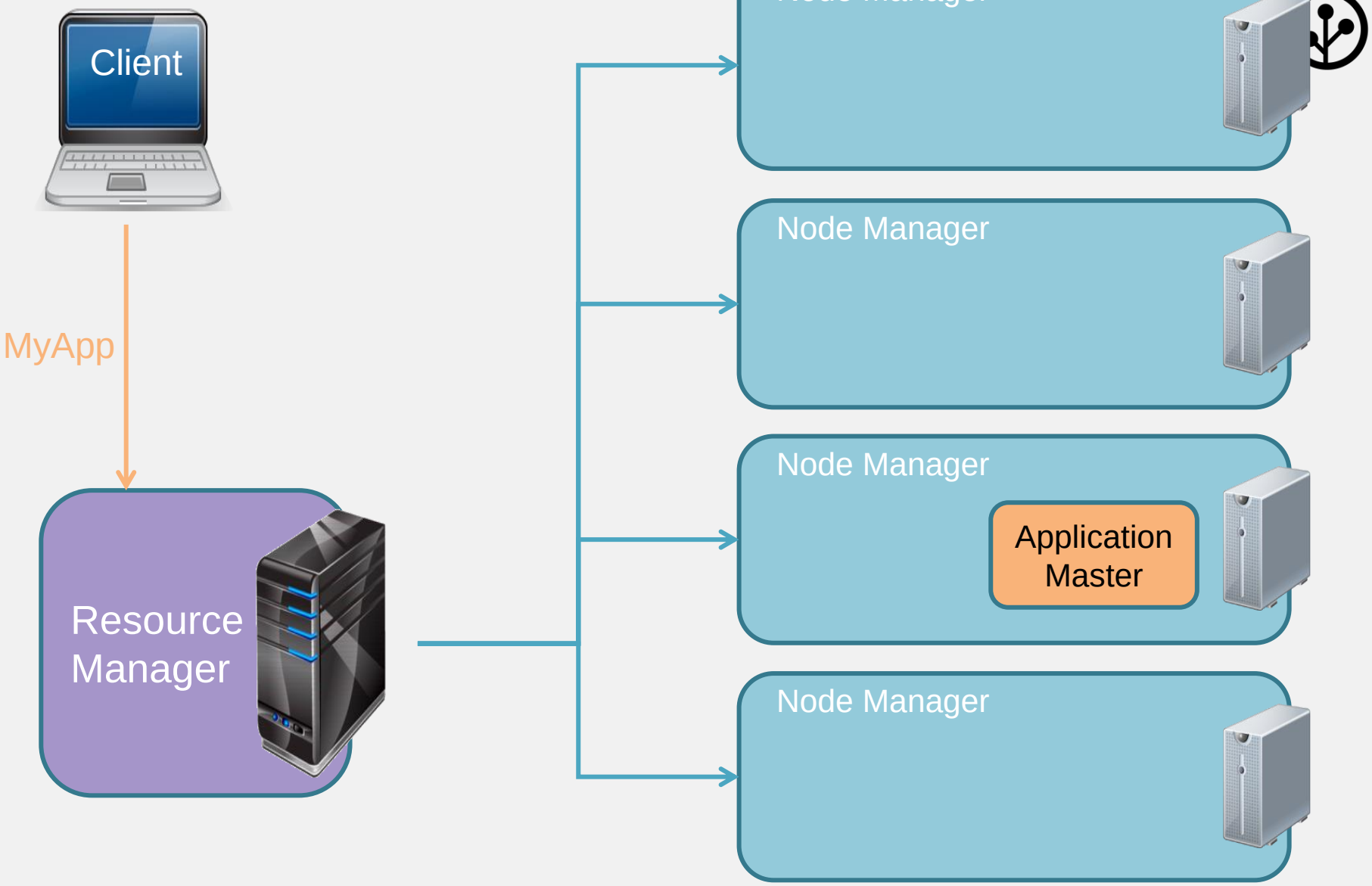
Компоненты YARN

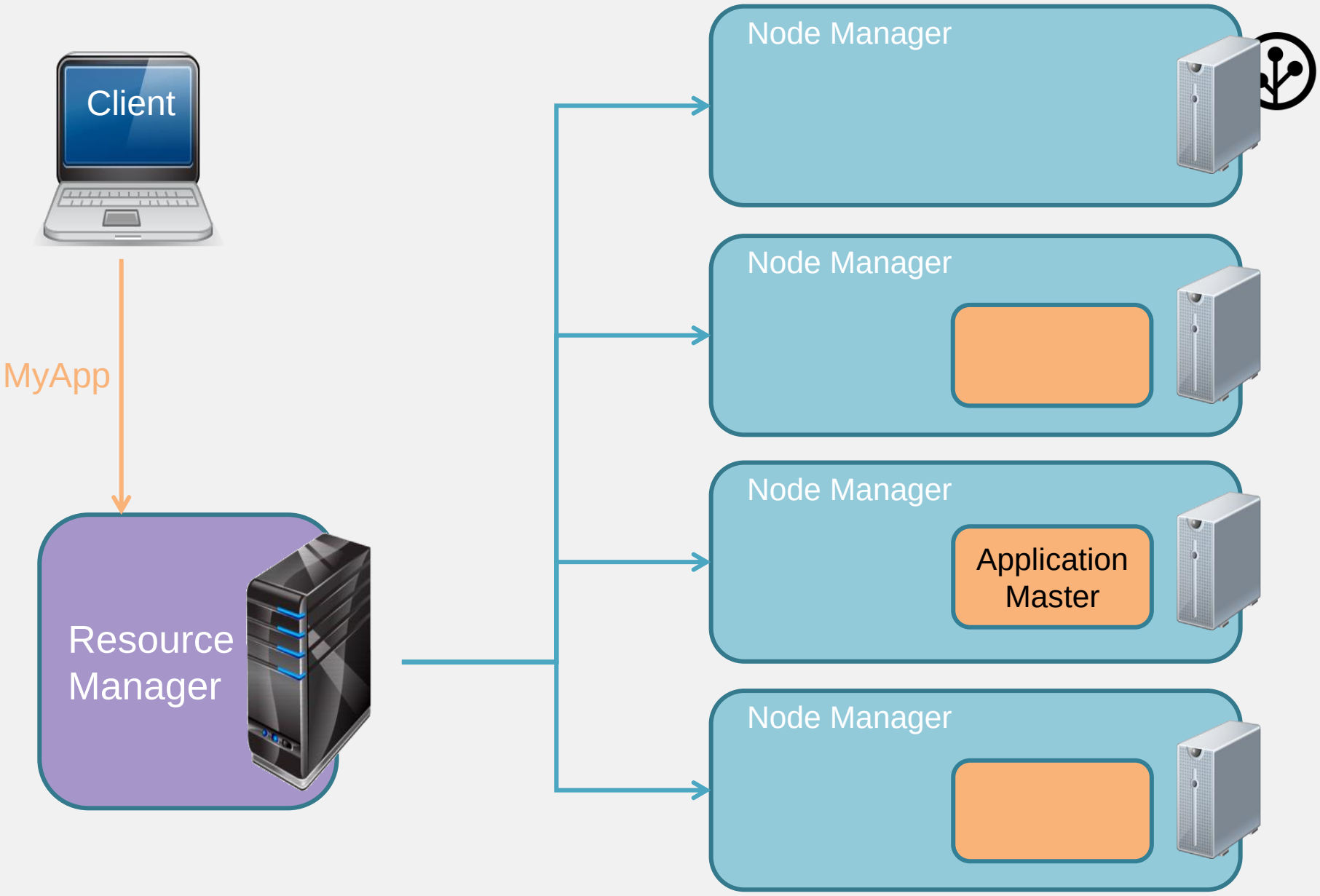
Application Master (AM)

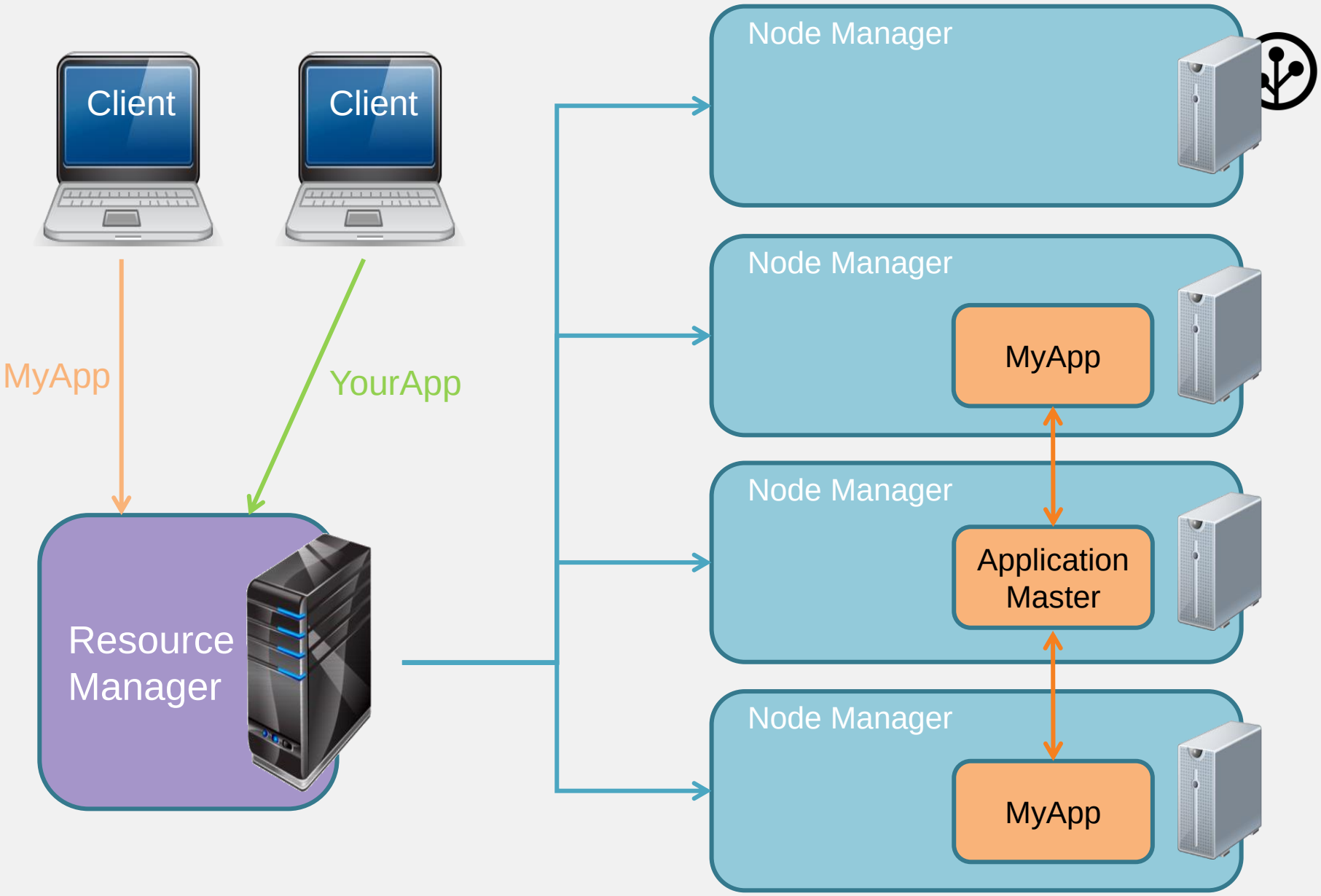
- Один для каждого приложения
- Зависит от типа задачи (свой для MR задач)
- Запускается в container
- Запрашивает другие containers для запуска приложения

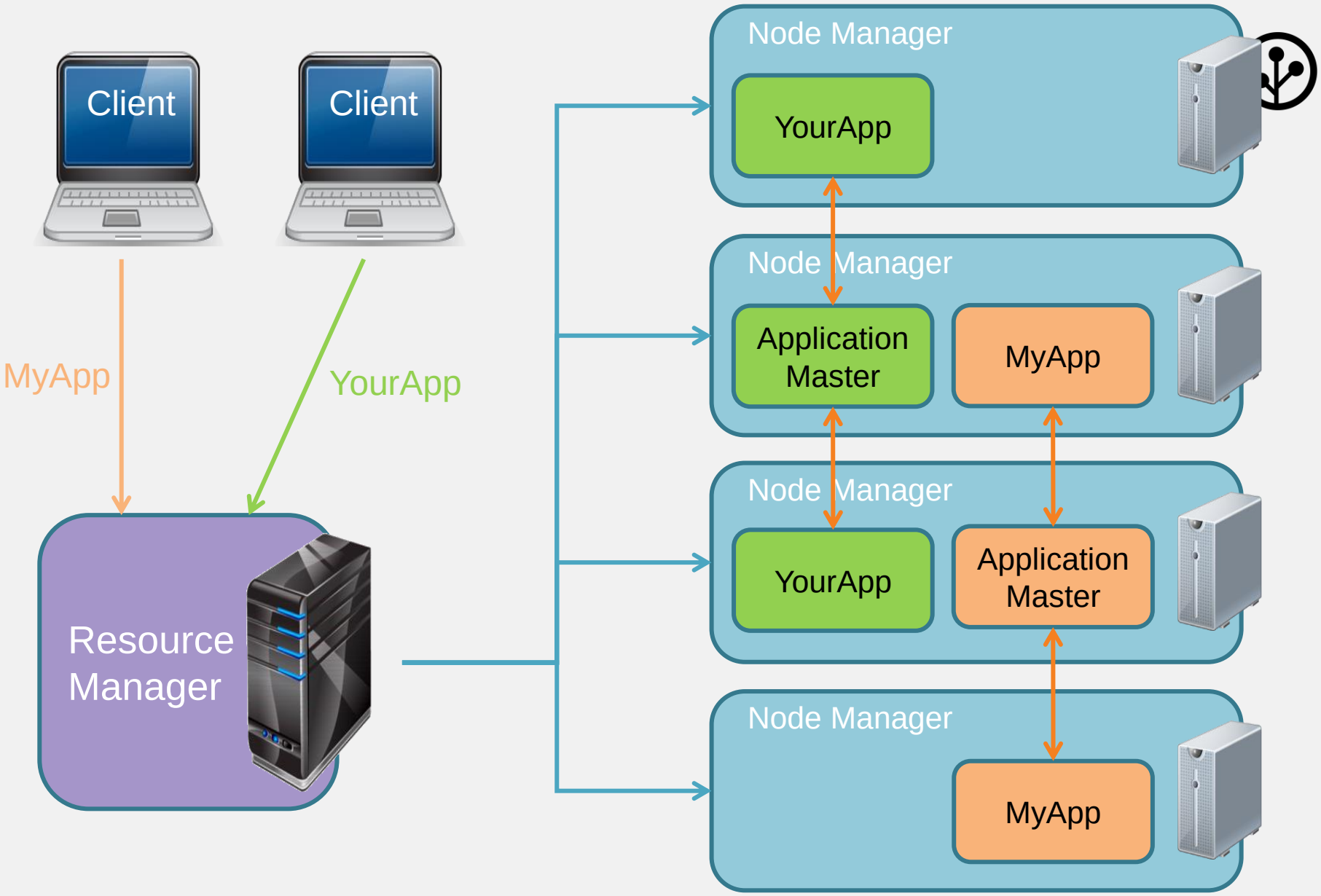



Client
Node Manager
Node Manager
Node Manager
Node Manager
Resource Manager


Client
MyApp
Resource Manager
Node Manager
Node Manager
Node Manager
Application Master
Node Manager


Client
MyApp
Resource Manager
Node Manager
Node Manager
Node Manager
Application Master
Node Manager


Client
Client
MyApp
YourApp
Resource Manager
Node Manager
Node Manager
MyApp
Node Manager
Application Master
Node Manager
MyApp


Client
Client
MyApp
YourApp
Resource Manager
Node Manager
YourApp
Node Manager
Application Master
MyApp
Node Manager
YourApp
Application Master
Node Manager
MyApp

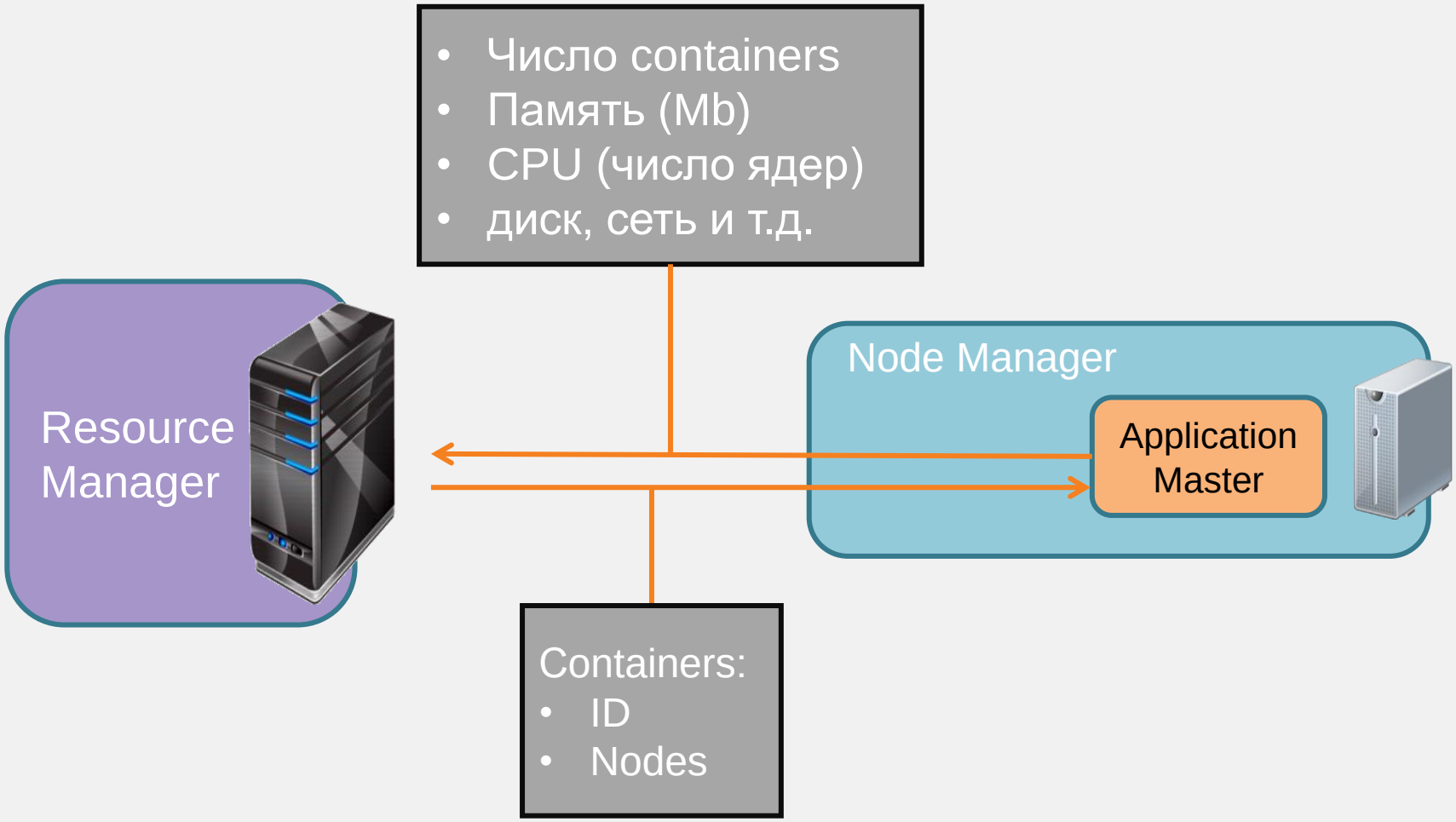
Функции Resource Manager

- Управляет нодами
- Управляет containers
- Взаимодействует с Application Manager
- Отвечает за защиту данных

Функции Node Manager

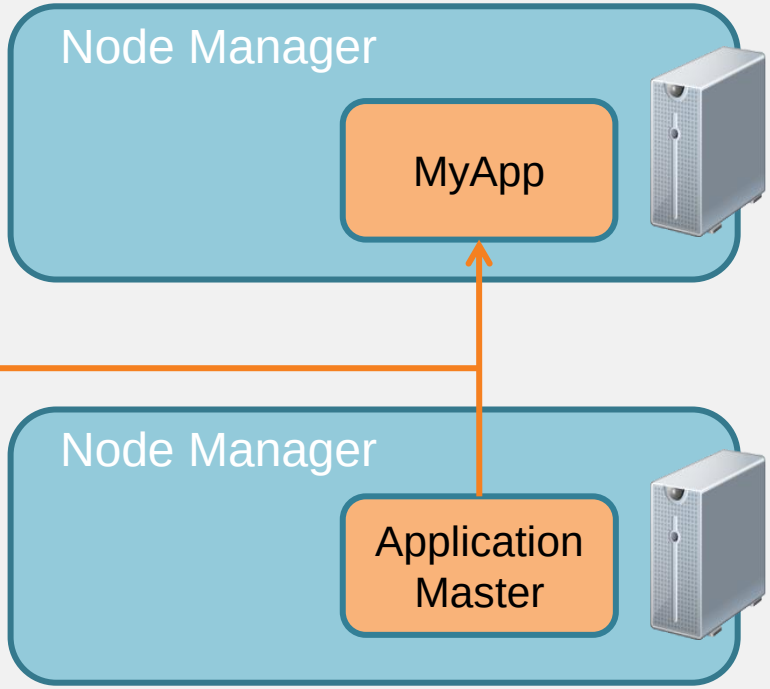
- Взаимодействует с RM
- Управляет процессами в контейнере
- Логирует данные приложений
- Поддерживает защиту ACL на уровне ноды

Запрос ресурсов



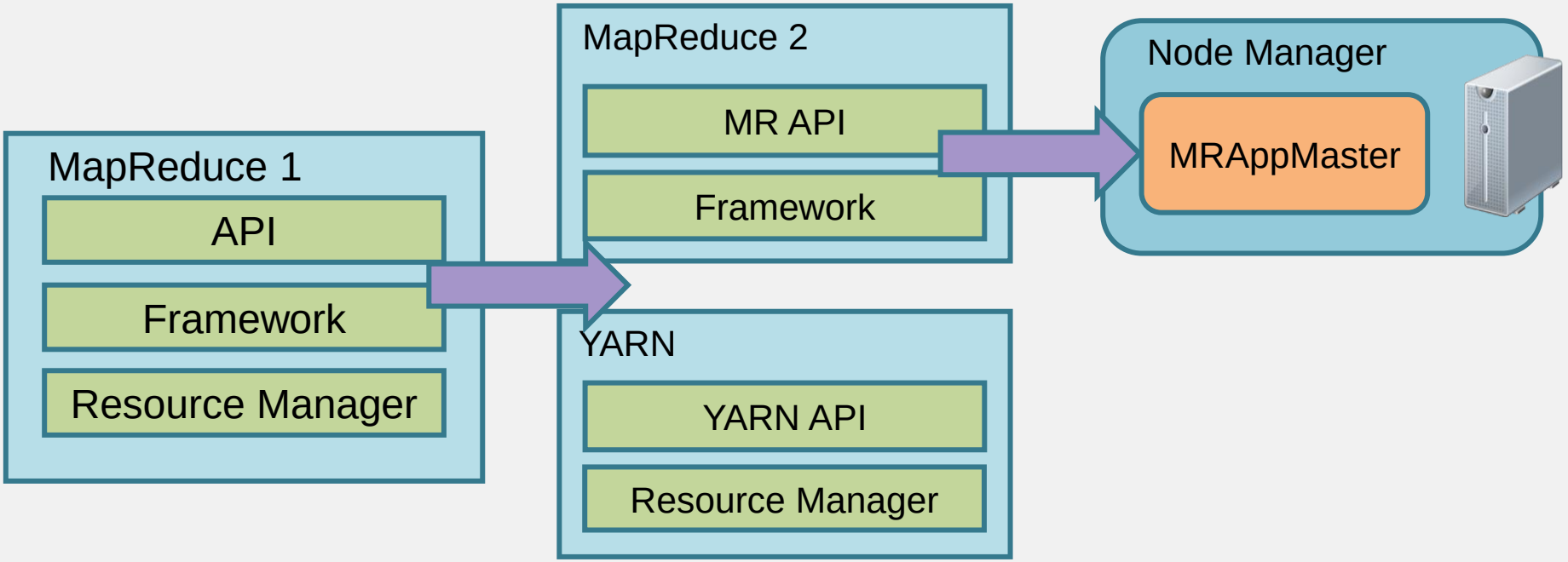
Запуск container

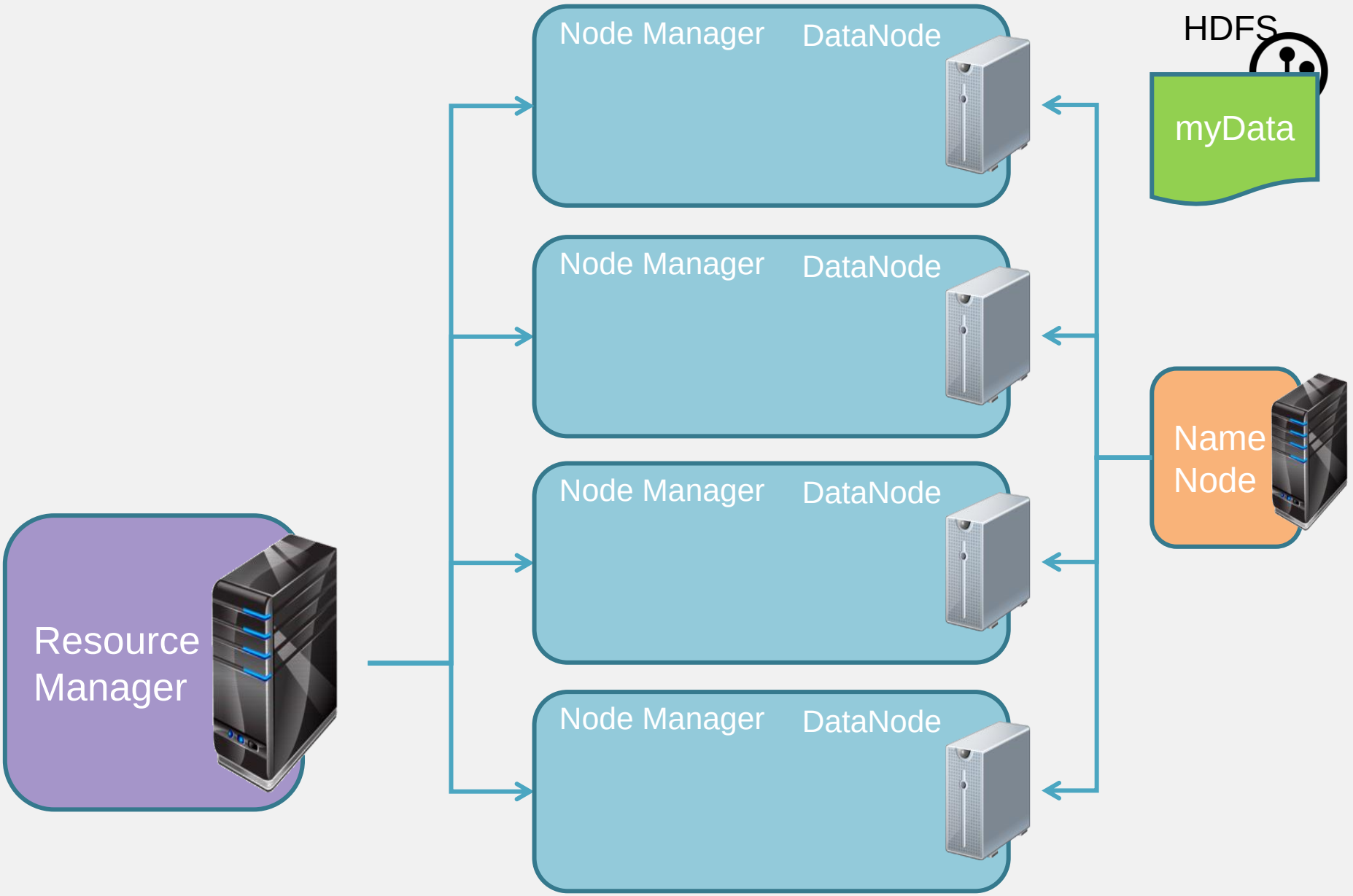
- Container ID
- Команды (для запуска приложения)
- Окружение (конфигурация)
- Локальные ресурсы (код задачи, HDFS файлы)



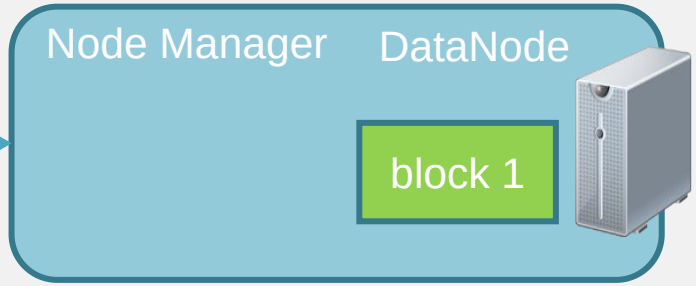
YARN и MapReduce

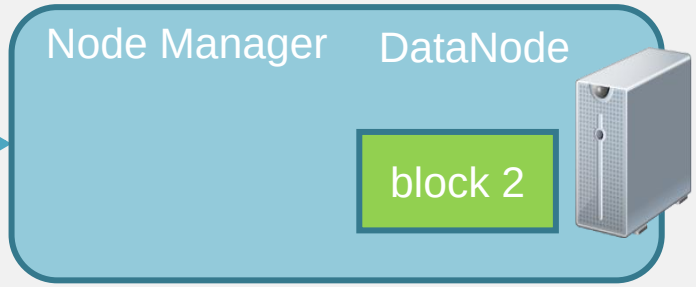
- YARN не знает какой тип приложения он запускает
- MRAppMaster – AM для MapReduce



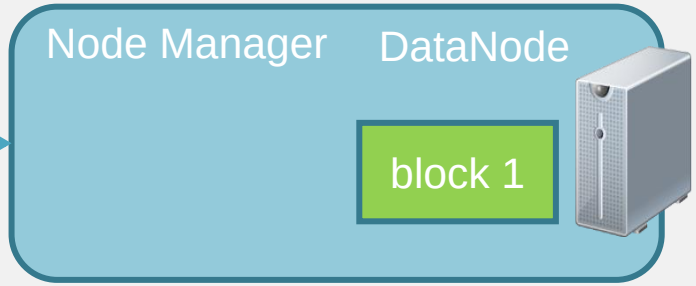

HDFS
myData
Node Manager
DataNode
Node Manager
DataNode
Node Manager
DataNode
Node Manager
DataNode
Name Node
Resource Manager

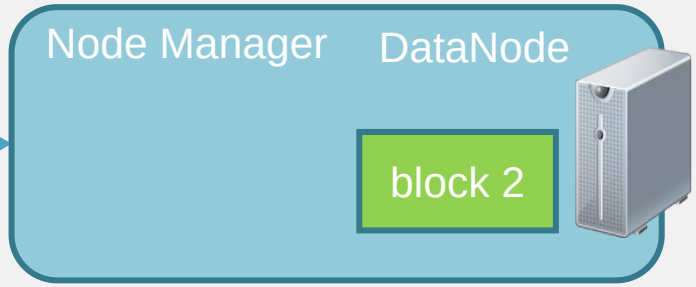
```
$ hadoop jar wc.jar
WordCount mydata output
```

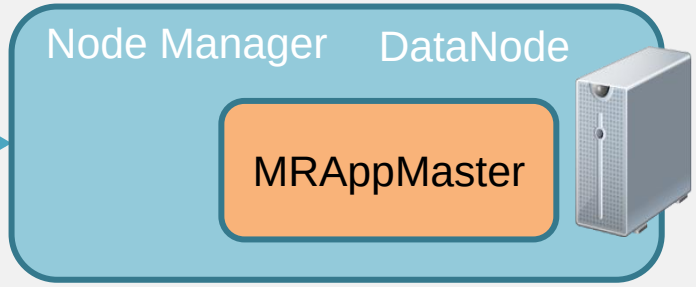




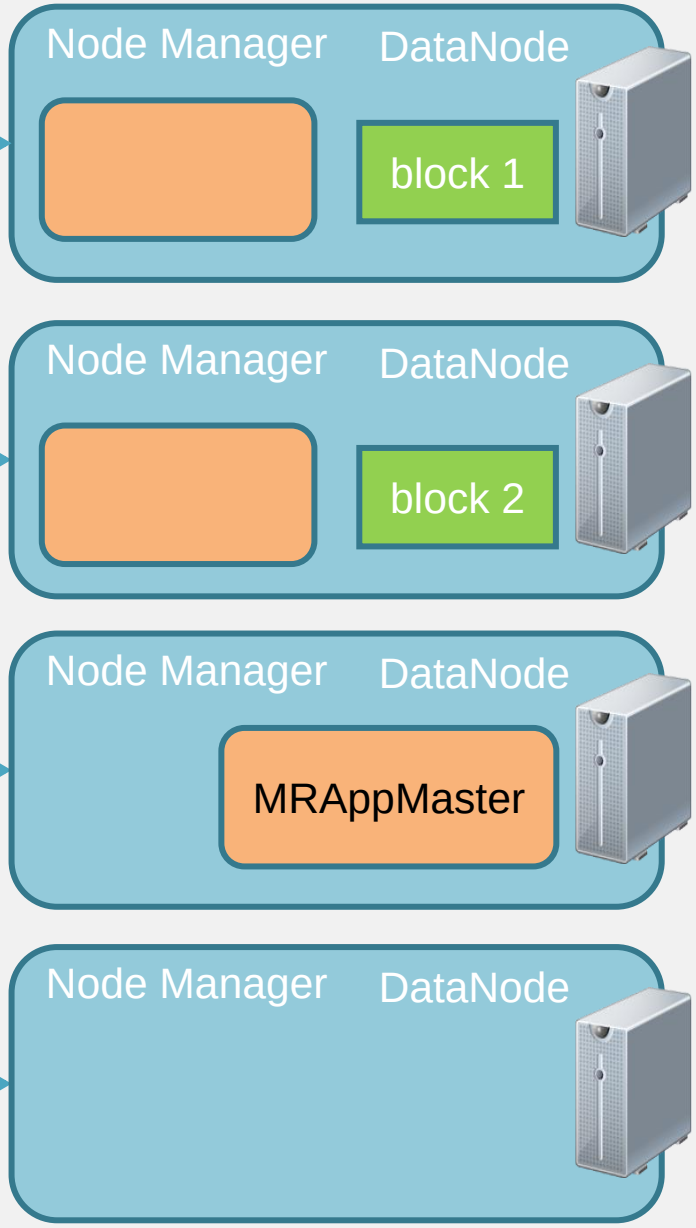
```
$ hadoop jar wc.jar
WordCount mydata output
```



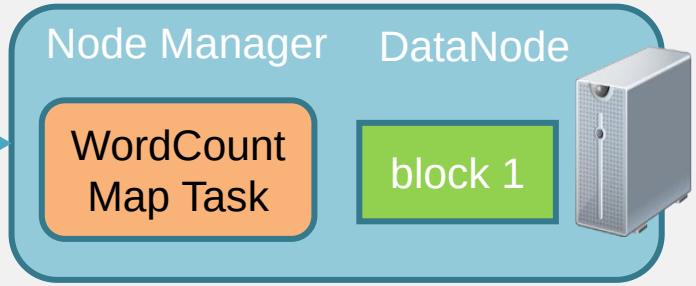


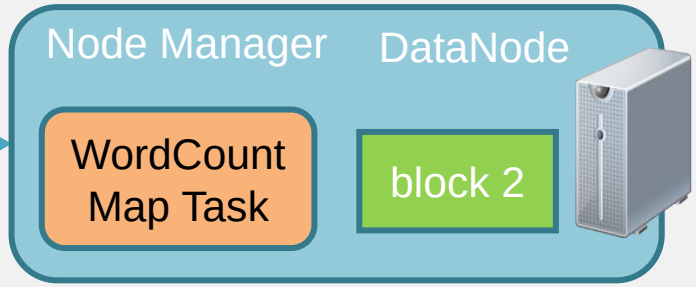


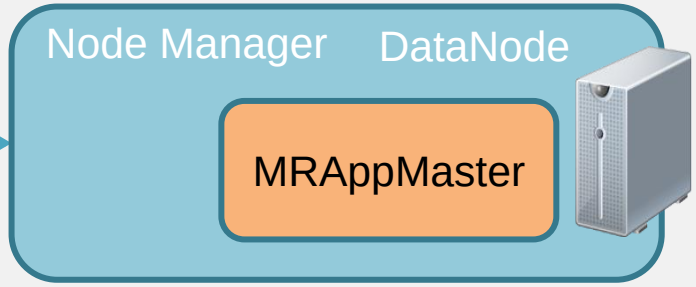
```
$ hadoop jar wc.jar
WordCount mydata output
```



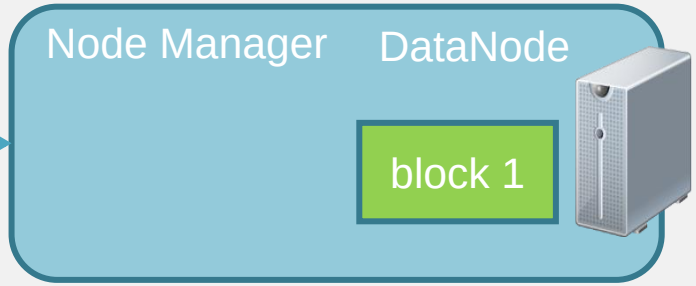
```
$ hadoop jar wc.jar
WordCount mydata output
```

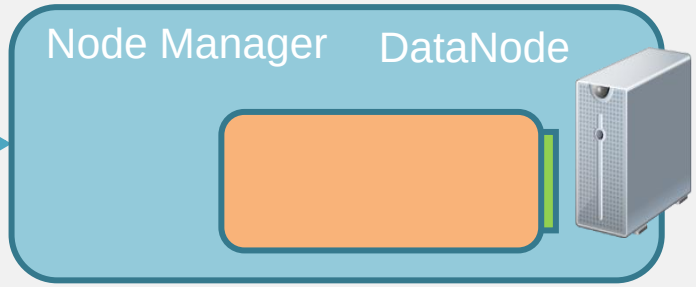


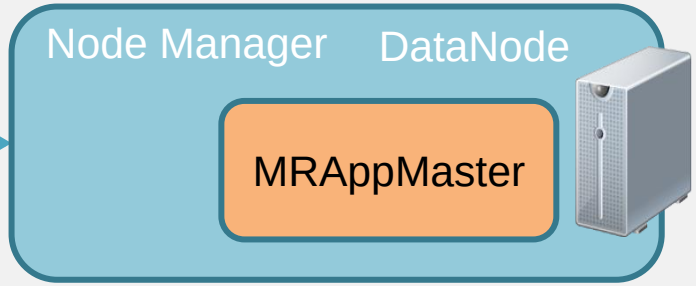


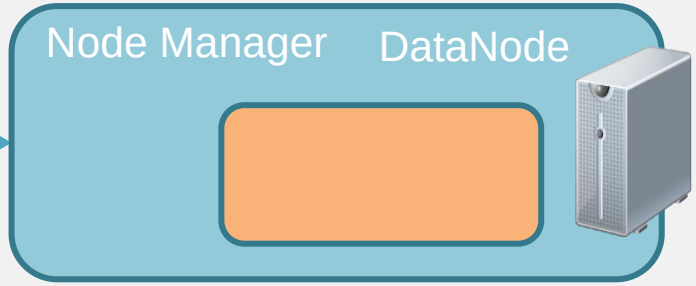


```
$ hadoop jar wc.jar
WordCount mydata output
```

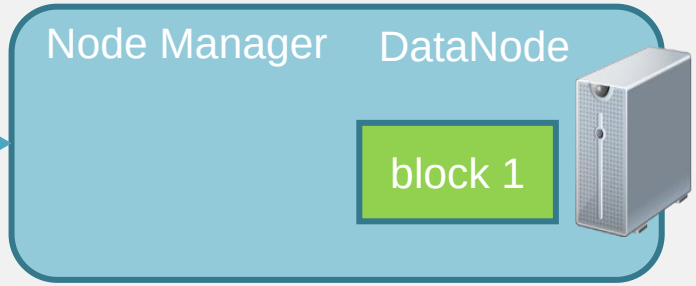


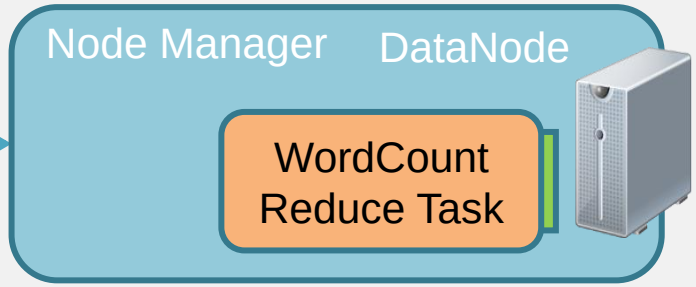


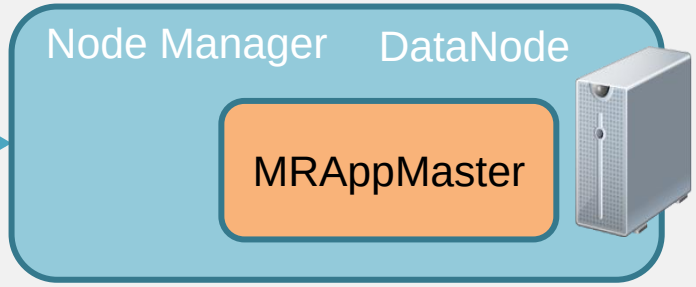


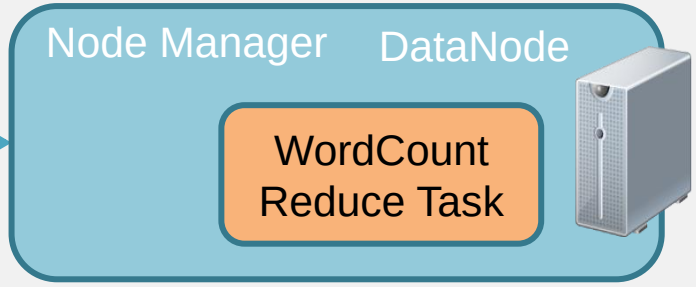


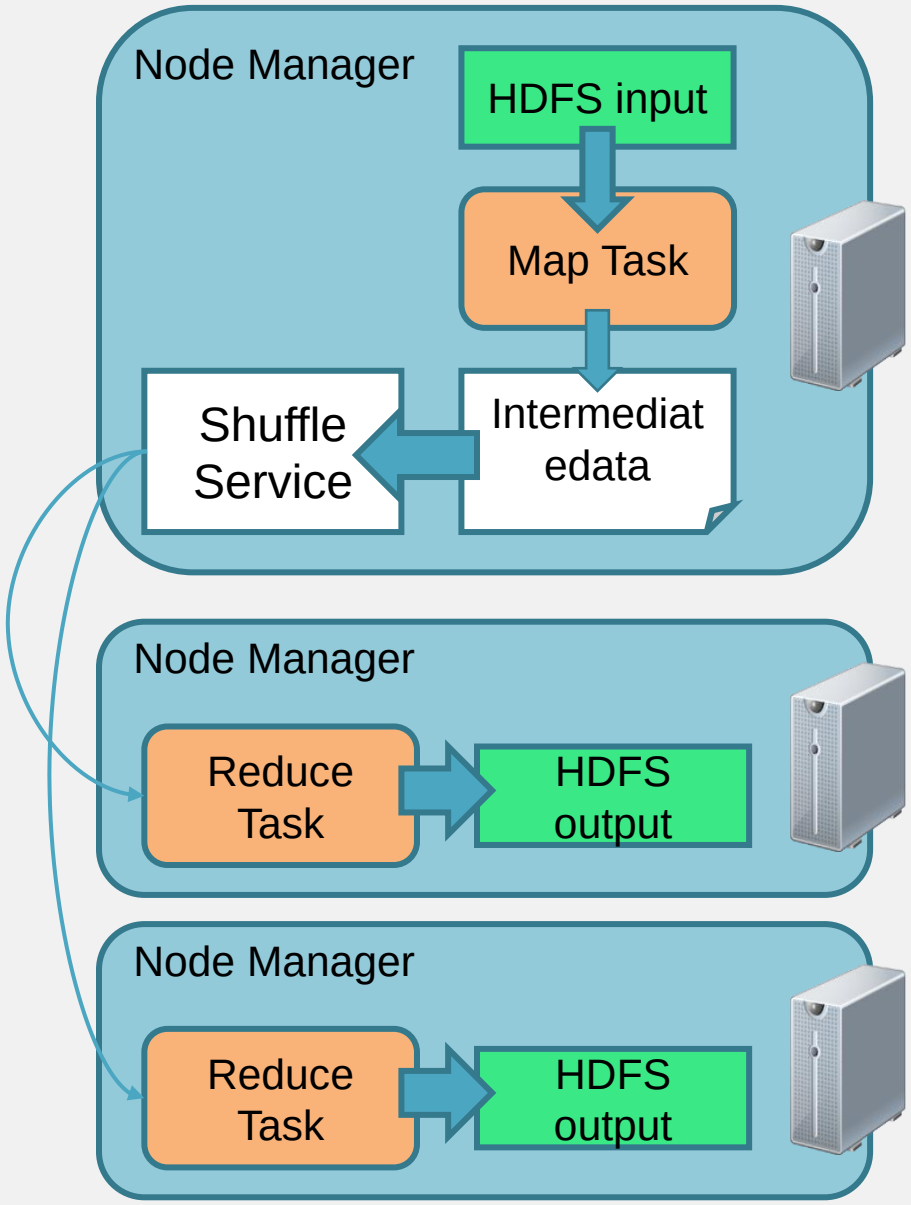
```
$ hadoop jar wc.jar
WordCount mydata output
```










Node Manager
HDFS input
Map Task
Intermediat
edata
Shuffle
Service
Node Manager
Reduce
Task
HDFS
output
Node Manager
Reduce
Task
HDFS
output

Uber задачи

- **MRAppMaster** запускает **map** и **reduce** внутри одной JVM
- Условия:
  - Меньше 10 мапперов
    - *mapreduce.job.ubertask.maxmaps*
  - Один редьюсер
    - *mapreduce.job.ubertask.maxreduces*
  - Размер входных данных меньше размера 1 HDFS-блока
    - *mapreduce.job.ubertask.maxbytes*
- Запуск uber-задач можно отключить
  - *mapreduce.job.ubertask.enable=false*

Управление памятью в YARN

Resource Manager выделяет память:

- минимум:

  yarn.scheduler.minimum-allocation-mb *(1024 Mb)*

- максимум:

  yarn.scheduler.maximum-allocation-mb *(8192 Mb)*

Управление памятью в YARN

- Каждый Node Manager имеет лимит на размер выделенной памяти:
  - ***yarn.nodemanager.resource.memory-mb*** *(8192 Mb)*
- Сумма всех containers не может превышать этот лимит
- Node Manager не создает container, если нет достаточно памяти

Управление памятью в YARN

- Размер containers для map и reduce:
  - ***mapreduce.map.memory.mb=1536***
  - ***mapreduce.reduce.memory.mb=3072***
- Размер JVM Heap:
  - ***mapreduce.map.java.opts=-Xmx1024m***
  - ***mapreduce.reduce.java.opts=-Xmx2560m***

Управление памятью в YARN

- Размер виртуальной памяти задается как коэффициент к физической памяти:
  - **yarn.nodemanager.vmem-pmem-ratio=*2.1***

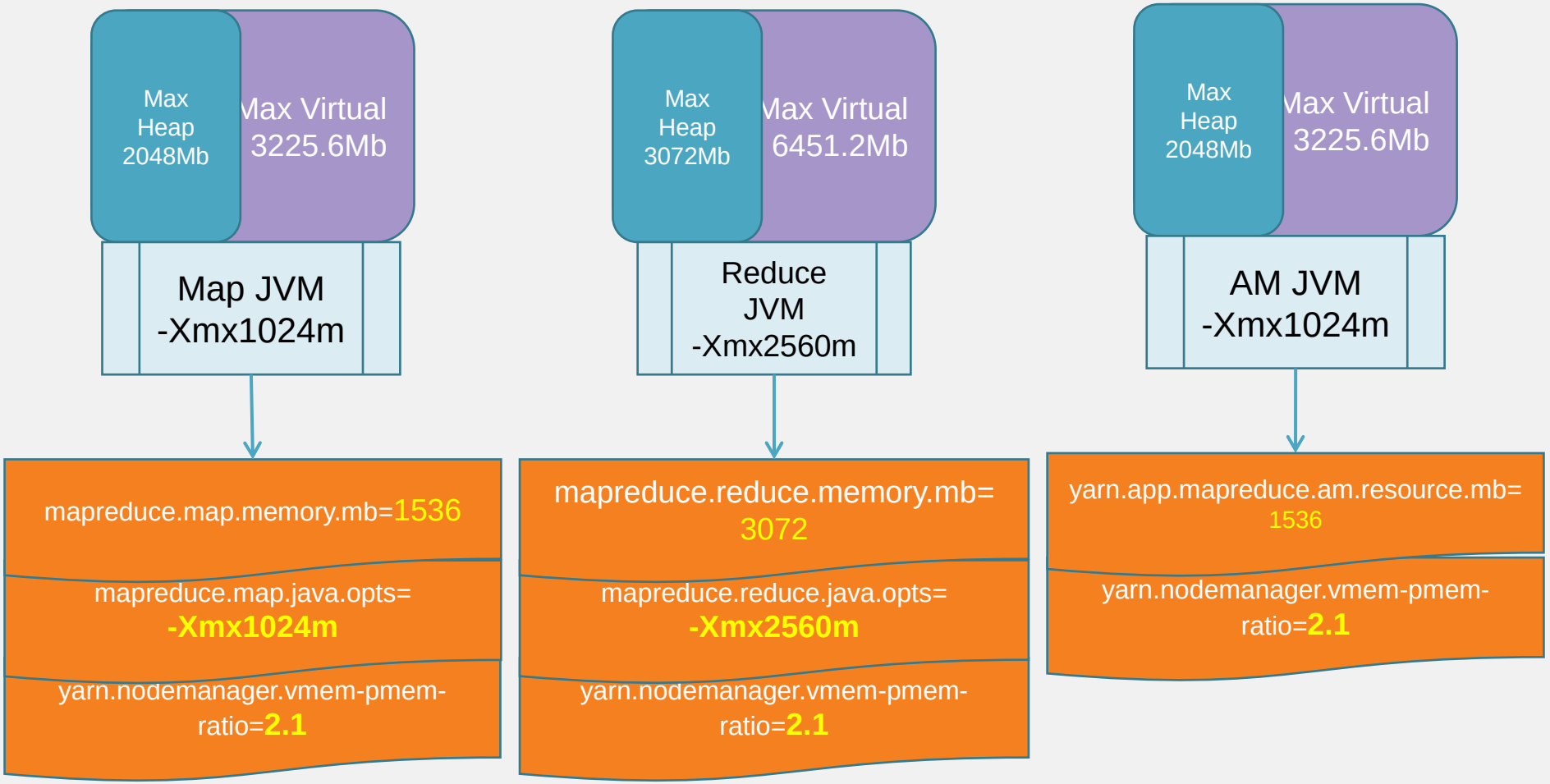
Управление памятью в YARN

Если **map** или **reduce** превышает лимит виртуальной или физической памяти, то **Node Manager** убивает соответствующий контейнер

```
Current usage: 2.1gb of 2.0gb physical memory used; 1.6gb
of 3.15gb virtual memory used. Killing container.
```

Resource Manager
yarn.scheduler.maximum-allocation-mb=**8192**
yarn.scheduler.minimum-allocation-mb=**1024**

Node Manager
yarn.nodemanager.resource.memory-mb=**24576**



Обработка сбоев

Сбои могут случится:

- Container (Task)
- Application Manager (MRAppMaster)
- Node Manager
- Resource Manager

Сбой в работе контейнера

- Что может случиться:
  - Исключительная ситуация (exception)
  - Падение JVM / выключение сервера
  - Зависание процесса
- MRAppMaster перезапускает задачу
- Число перезапусков определяется через:
  - *mapreduce.map.maxattempts*
  - *mapreduce.reduce.maxattempts*

Сбой в Application Manager

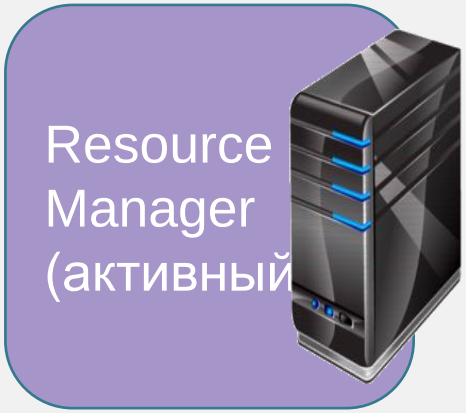
- Что может случиться:
  - Работа AM завершается с ошибкой
  - Сигналы от AM не доходят до RM
- Resource Master перезапускает AM
  - *yarn.resourcemanager.am.max-retries*=**2**
- MRAppMaster может восстановить состояние задачи до перезапуска:
  - *yarn.app.mapreduce.am.job.recovery.enable*=***true***

Сбой в Node Manager

- Что происходит:
  - NM перестает посылать сигналы RM
  - Контейнер на NM не посылает сигналы AM
- RM убирает NM из списка активных NM, если от него не приходит сигнал 10 минут
  - *yarn.resourcemanager.nm.liveness-monitor.expiryinterval-ms*
- MRAppMaster заносит NM в “черный список”, если произошло 3 падения контейнера
  - *mapreduce.job.maxtaskfailures.per.tracker*

Сбой в Resource Manager

- Никакие задачи и приложения не могут быть запущены
- Приводит к простою кластера



Использование YARN

- **Apache Giraph**
  - Итеративная система обработки графов
  - Facebook обрабатывает триллион ребер за ~4 мин
- **Spark**
  - Платформа для быстрой аналитики данных
- **Apache HAMA**
  - Фреймворк для массивных научных вычислений над матрицами, графами и сетями
- **Open MPI**

ТЕХНОСФЕРА

Отмечайтесь и оставляйте отзыв

Спасибо за внимание!

Евгений Чернов

e.chernov@corp.mail.ru