Лекция 3. Менеджер ресурсов YARN

Big Data Analytics:
Approaches and Tools

ИНСТИТУТ ЦИФРОВОГО ОБРАЗОВАНИЯ МГПУ

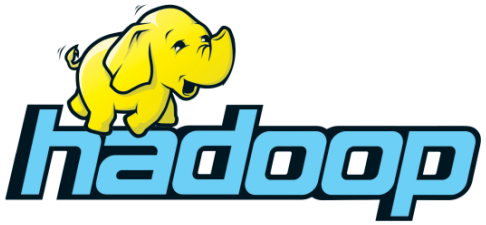
Основные темы

- Распределение ресурсов и мониторинг выполнения
- MapReduce 1
- YARN
- Запуск приложений
- Планирование выполнения

ИНСТИТУТ
ЦИФРОВОГО
ОБРАЗОВАНИЯ
МГПУ

Управление ресурсами

Управление ресурсами



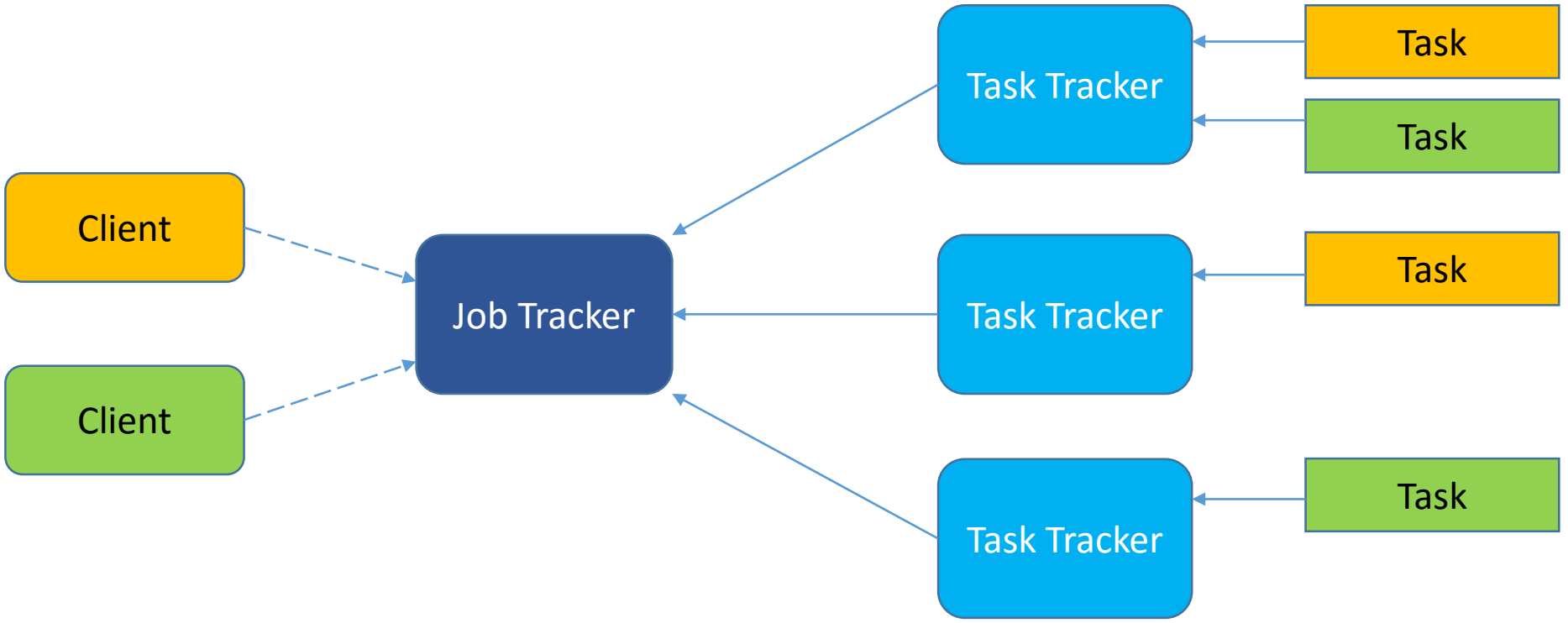
YARN

Зачем?

ИНСТИТУТ
ЦИФРОВОГО
ОБРАЗОВАНИЯ
МГПУ

MapReduce 1

Задачи MapReduce 1



Задачи MapReduce 1

JobTracker координирует все job’ы, запущенные в системе; планирует размещение на узлах с TaskTracker; отслеживает выполнение всех job’ов; хранить историю

TaskTracker запускает task и отправляет отчеты о выполнении JobTracker’у

Если выполнение task закончилось неудачей, то JobTracker может повторно запустить её на другом узле

ИНСТИТУТ ЦИФРОВОГО ОБРАЗОВАНИЯ МГПУ

Задачи MapReduce 1

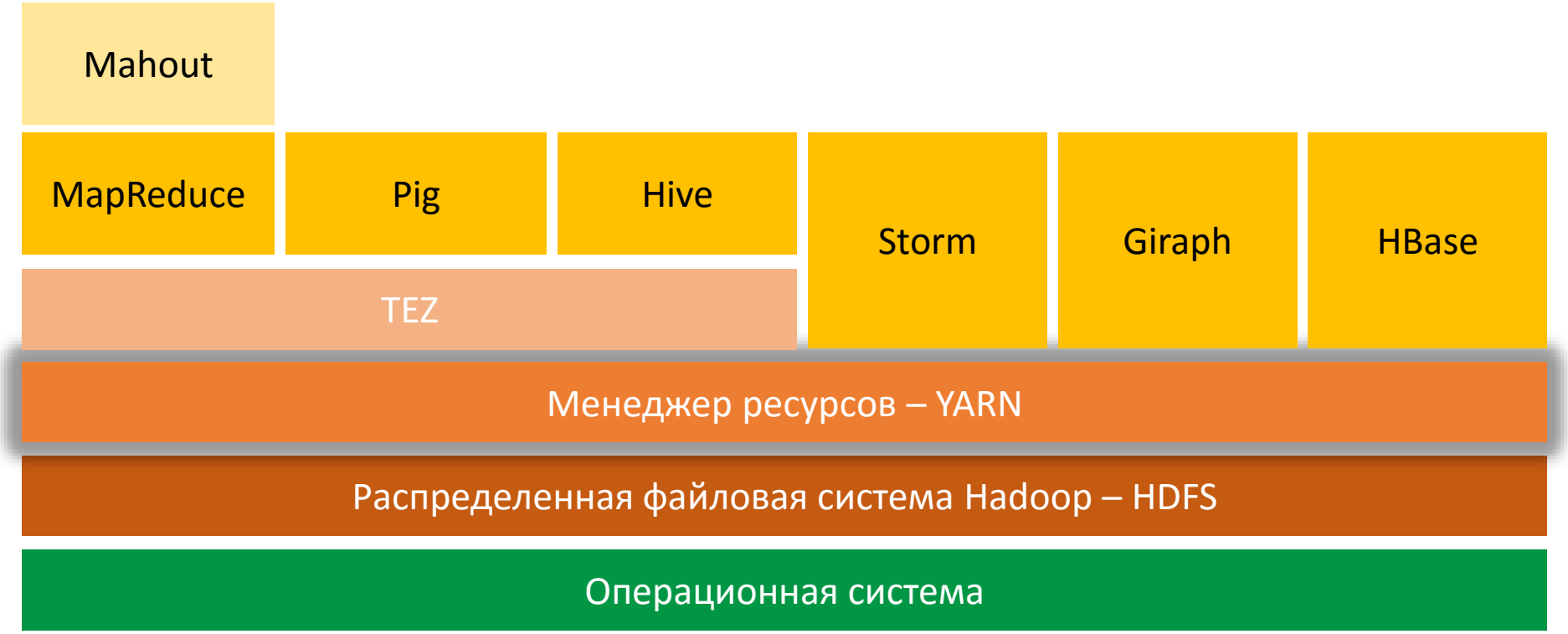
- MapReduce API
- MapReduce framework
- MapReduce system

ИНСТИТУТ
ЦИФРОВОГО
ОБРАЗОВАНИЯ
МГПУ

YARN - Yet Another Resource Negotiator

ИНСТИТУТ ЦИФРОВОГО ОБРАЗОВАНИЯ МГПУ

Стек Hadoop



Сравнение Job Tracker MapReduce 1 и YARN

MapReduce1

YARN

Управление ресурсами кластера

Управление приложением

ИНСТИТУТ ЦИФРОВОГО ОБРАЗОВАНИЯ МГПУ

Особенности YARN

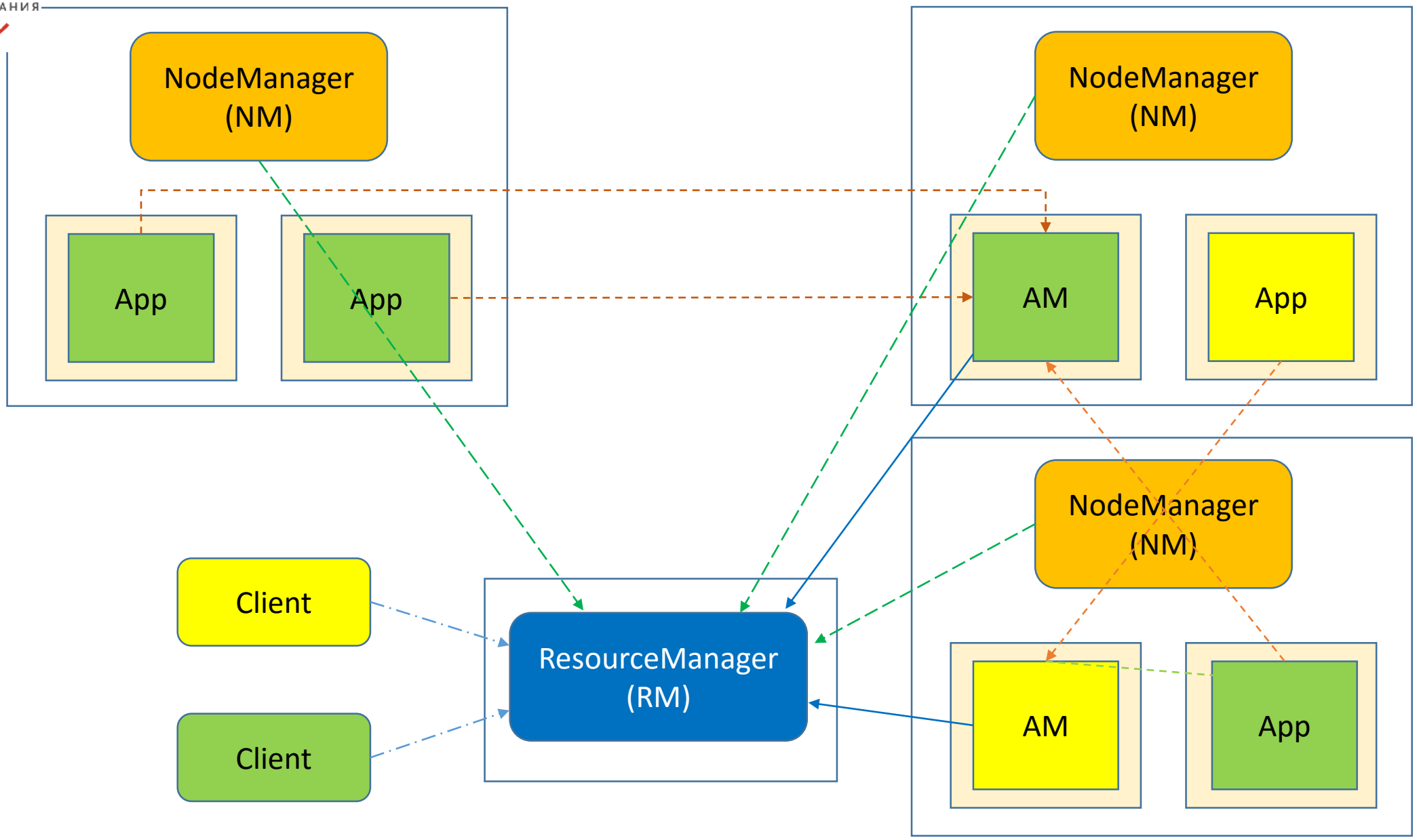
- Масштабируемость — 10000 узлов, 100000 задач (tasks)
- Доступность
- Гибкое использование ресурсов
- Множество пользователей
- Возможность развертывания приложений различных платформ (MapReduce, Spark)

ИНСТИТУТ ЦИФРОВОГО ОБРАЗОВАНИЯ МГПУ

Словарь YARN

- Job
- Task
- ResourceManager
- NodeManager
- ApplicationMaster
- Container

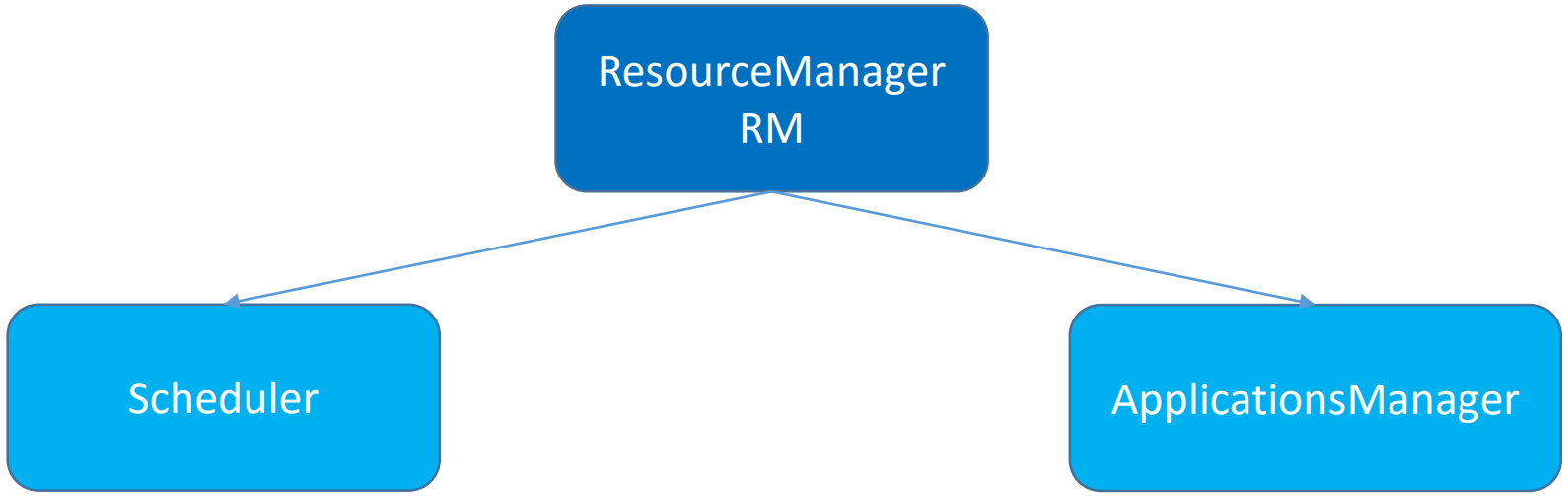
Архитектура YARN



ИНСТИТУТ
ЦИФРОВОГО
ОБРАЗОВАНИЯ
МГПУ

Job vs Task

ResourceManager



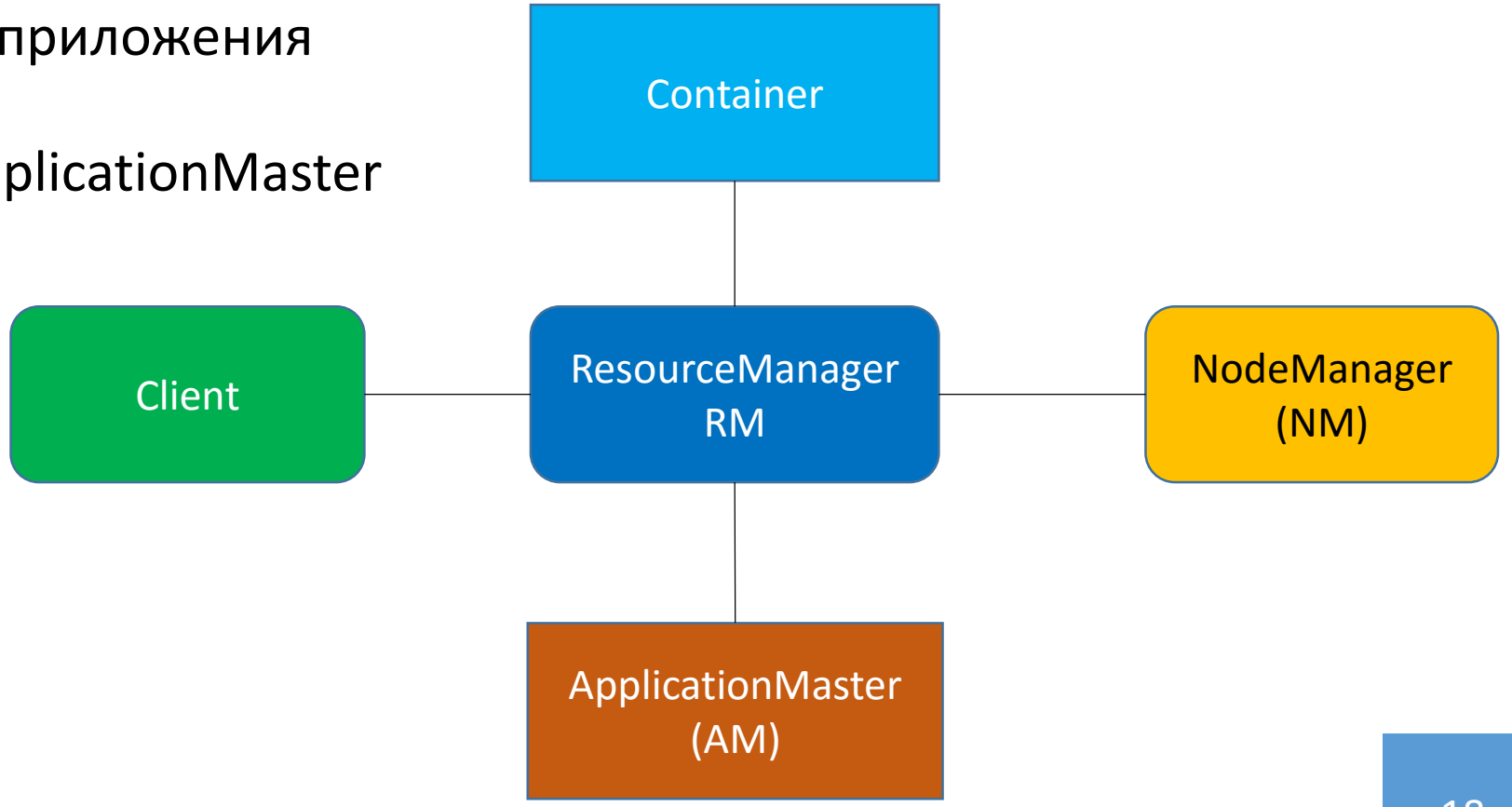
Отвечает за размещение ресурсов с учетом ограничений кластера и требований приложения

Отвечает за запуск приложений, запускает первый контейнер для ApplicationMaster

ResourceManager

- Запуск приложения клиента

- Мониторинг ресурсов
- Планирование запуска приложения
- Запуск container для ApplicationMaster



ИНСТИТУТ ЦИФРОВОГО ОБРАЗОВАНИЯ МГПУ

NodeManager

NodeManager управляет ресурсами узла:

- Отслеживает состояние узлов
- Отслеживает используемые ресурсы
- Управляет контейнерами

ApplicationMaster

ApplicationMaster отвечает за

- получение контейнеров с нужными количеством ресурсов для запуска приложения
- отслеживает статус контейнеров
- отслеживает выполнение приложений

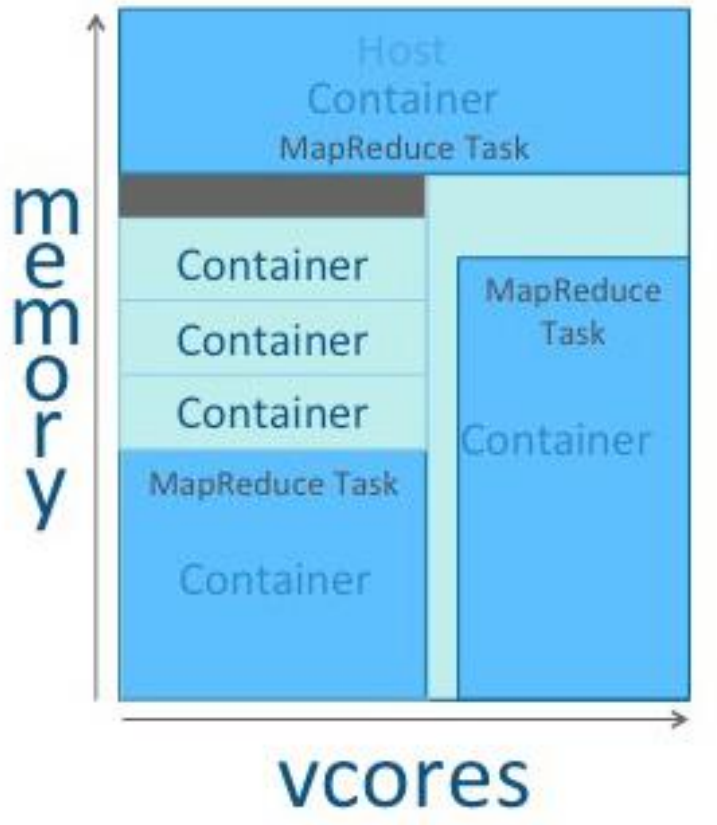
Container

Контейнер (container) – изолированная среда выполнения с определенным количеством вычислительных ресурсов (RAM, vCPU)

В контейнере выполняется задача приложения пользователя

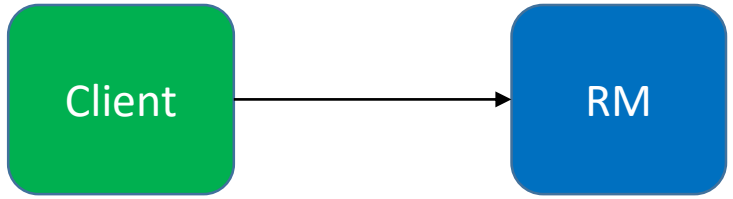
Непосредственно запускается NodeManager’ом по команде от ResourceManager или ApplicationMaster

При запуске подгружаются файлы выполняемой задачи (jar-файлы), конфигурация, общие ресурсы и пр.



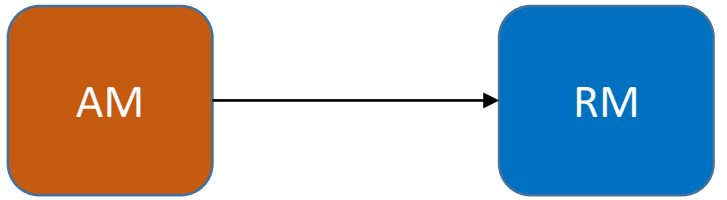
Протоколы взаимодействия

ApplicationClientProtocol



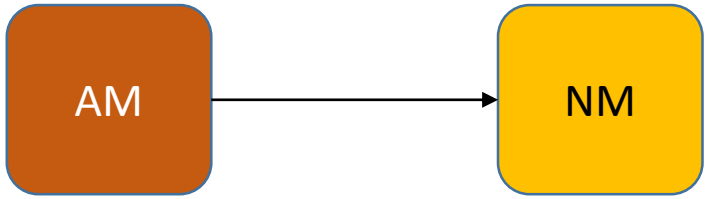
- submitApplication()
- forceKillApplication()

ApplicationMasterProtocol



- allocate()
- registerApplicationMaster()
- finishApplicationMaster()

ContainerManagementProtocol

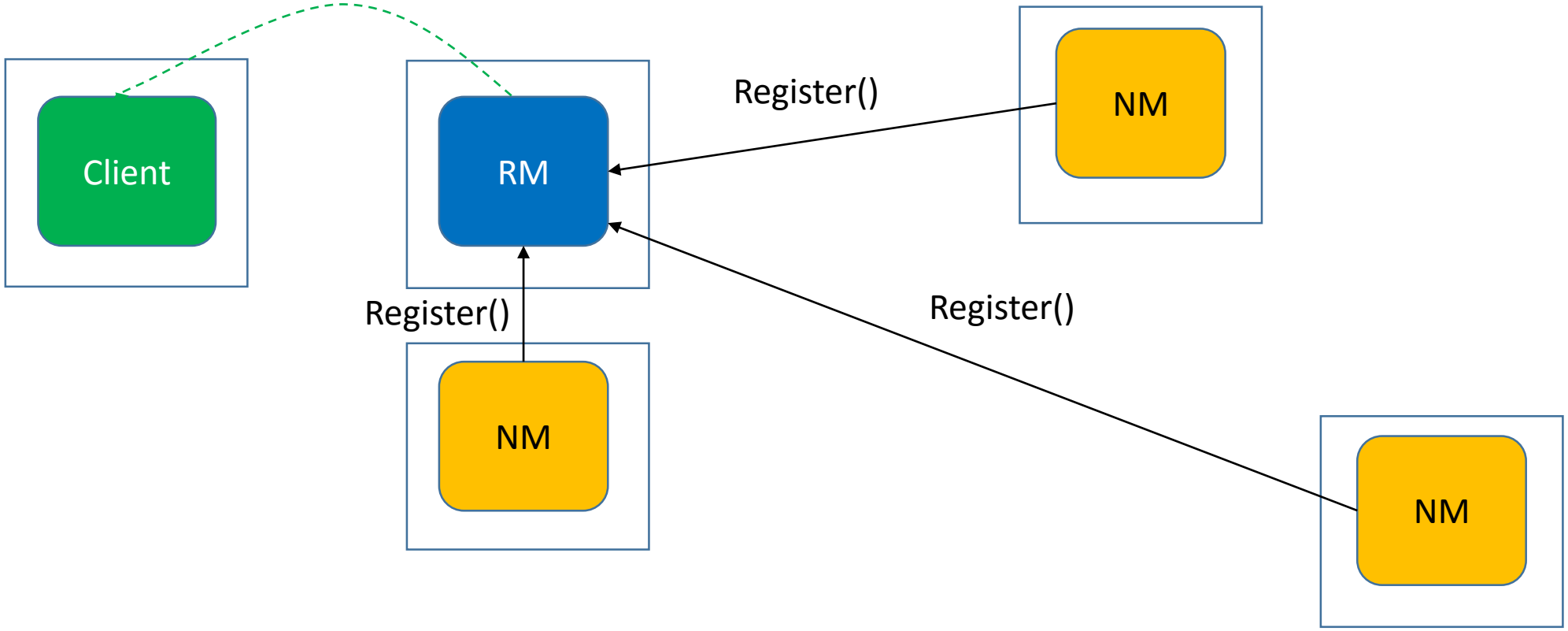


- startContainers()
- stopContainers()
- getContainerStatuses()

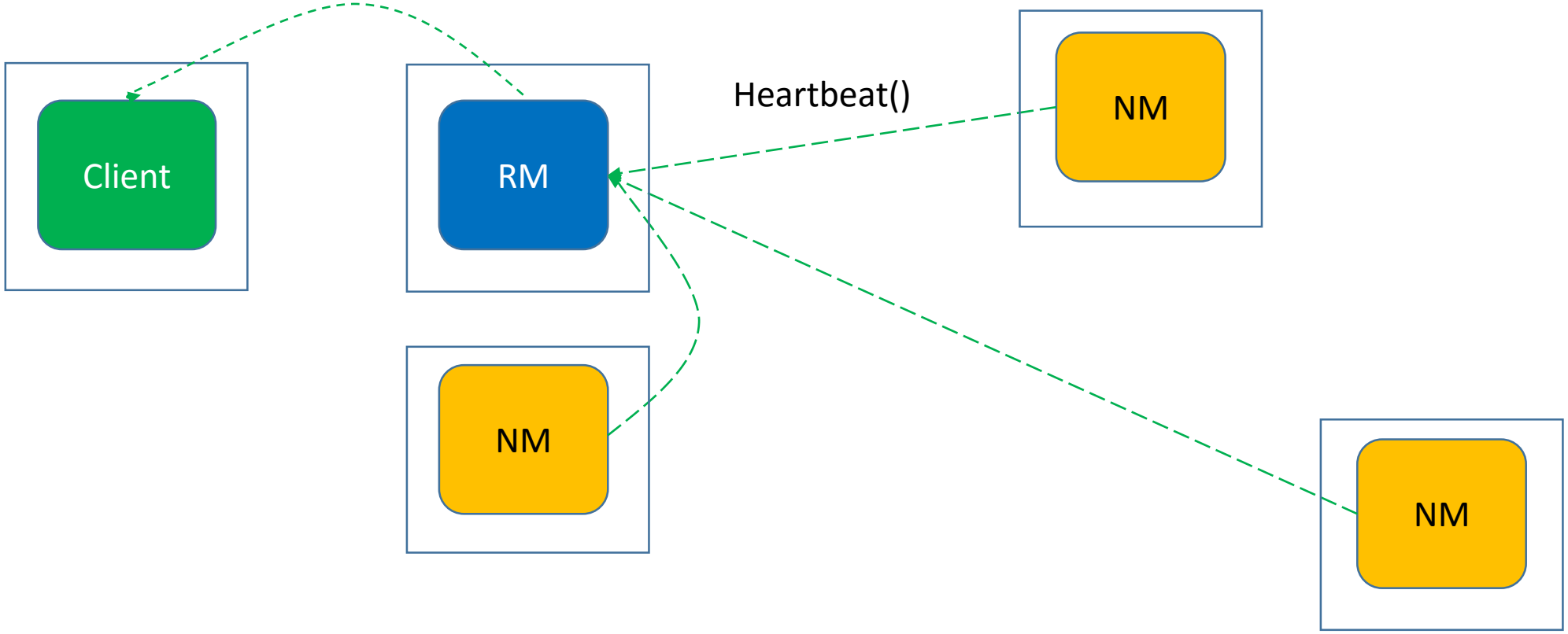
ИНСТИТУТ
ЦИФРОВОГО
ОБРАЗОВАНИЯ
МГПУ

Запуск приложения

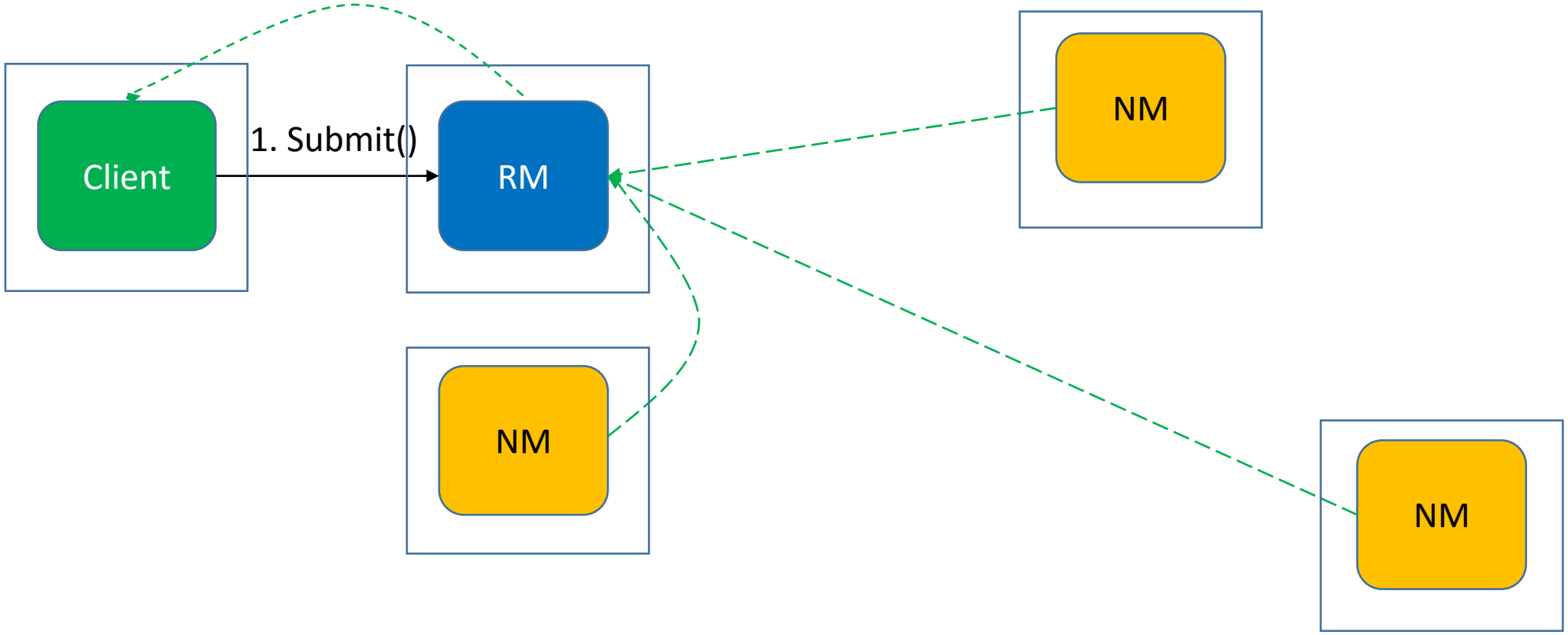
Запуск приложения



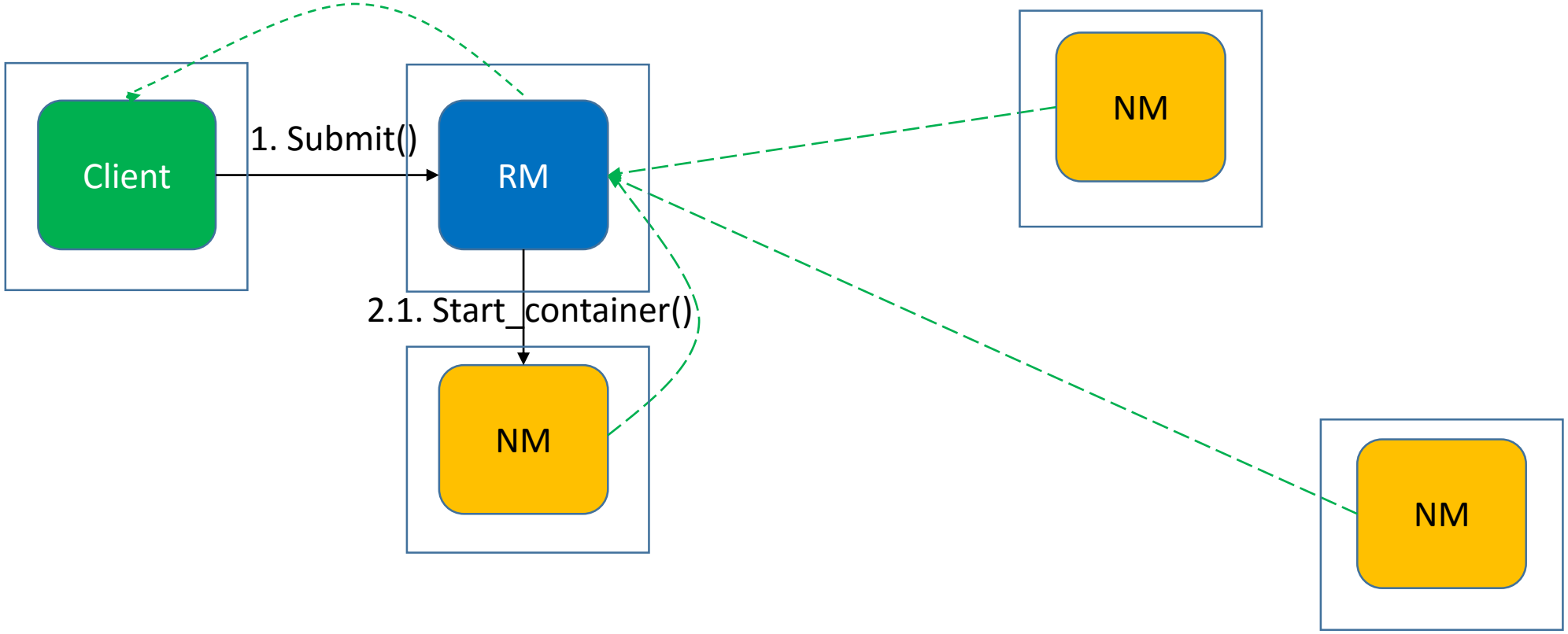
Запуск приложения



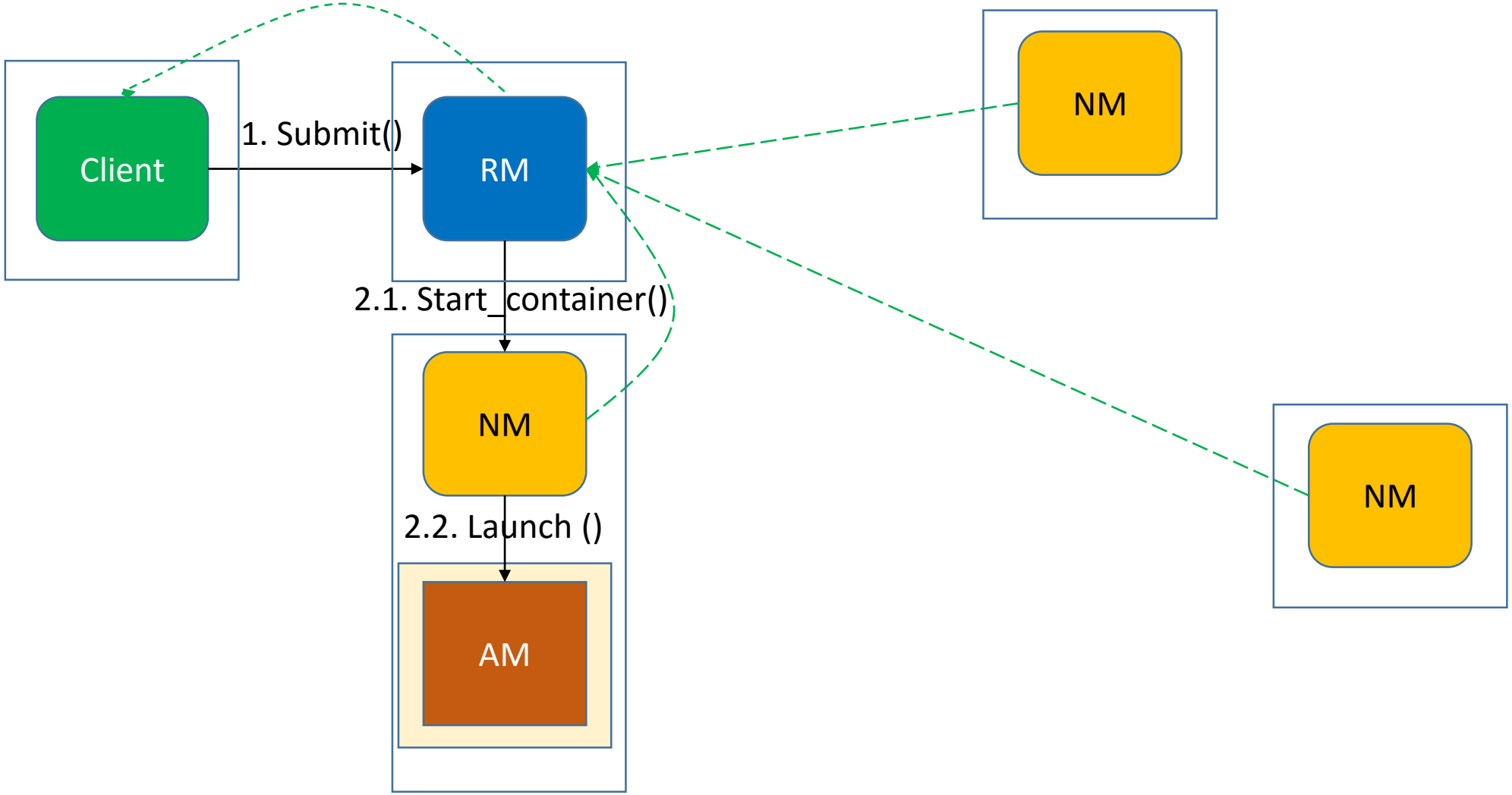
Запуск приложения



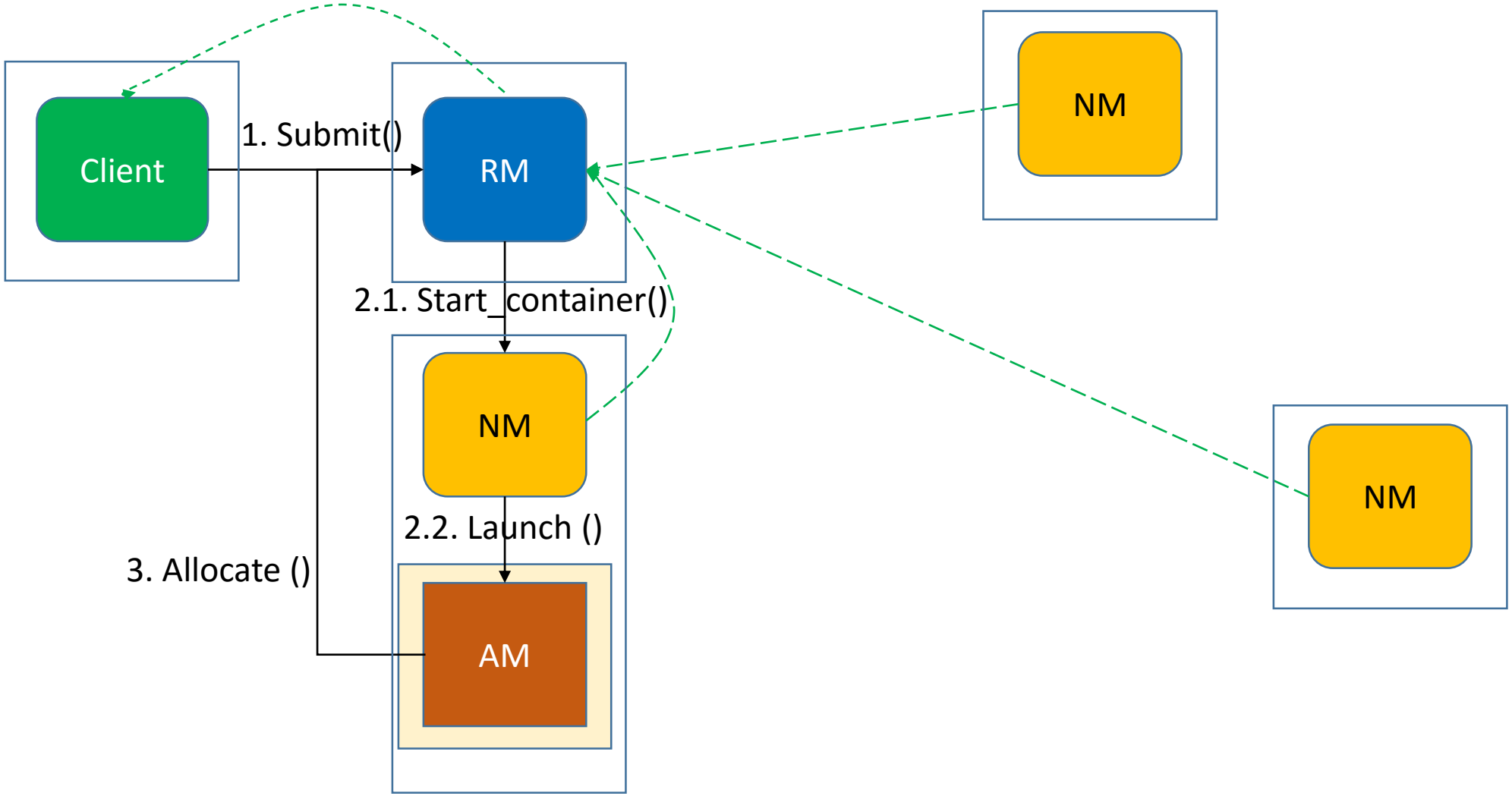
Запуск приложения



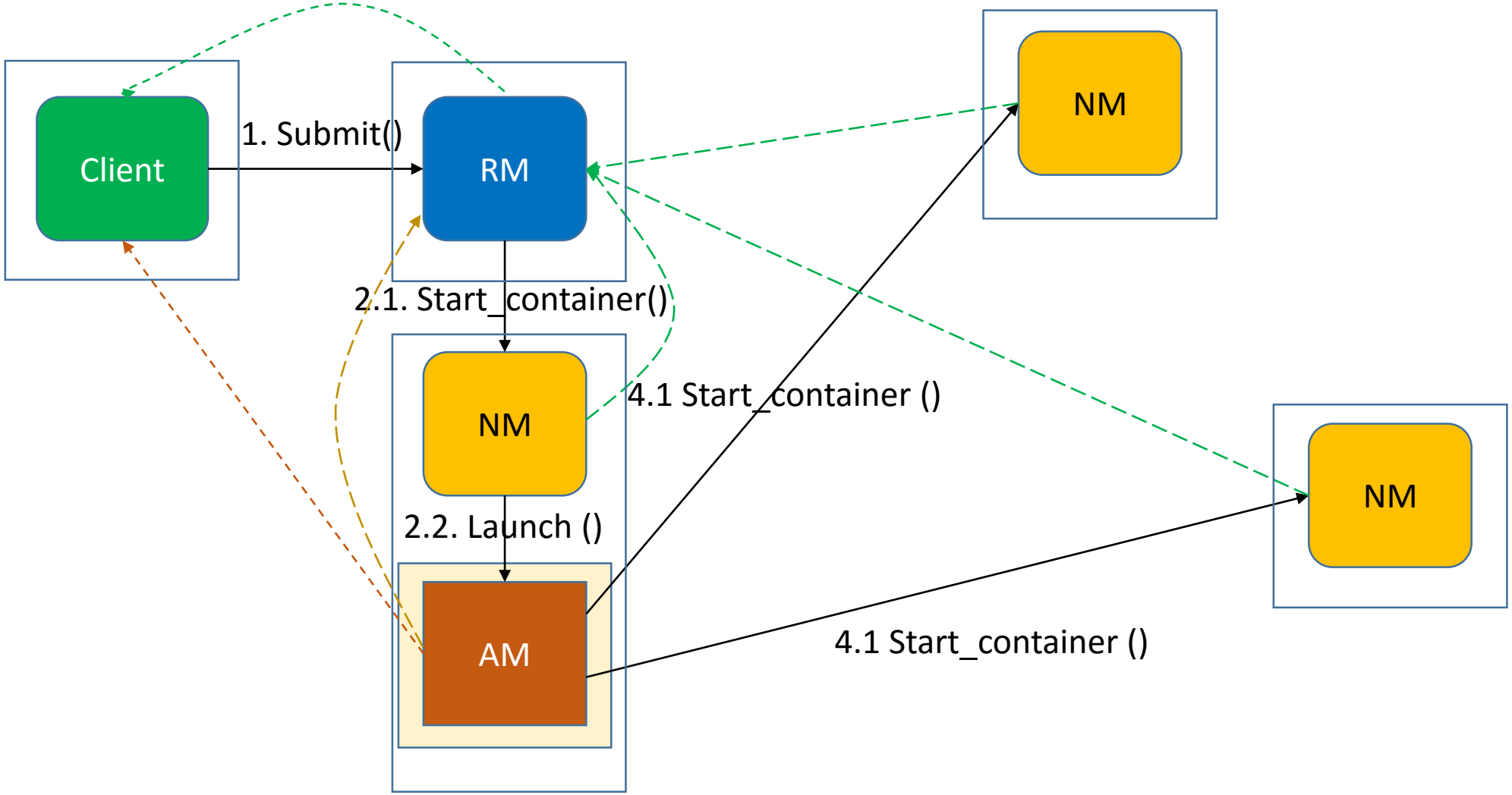
Запуск приложения



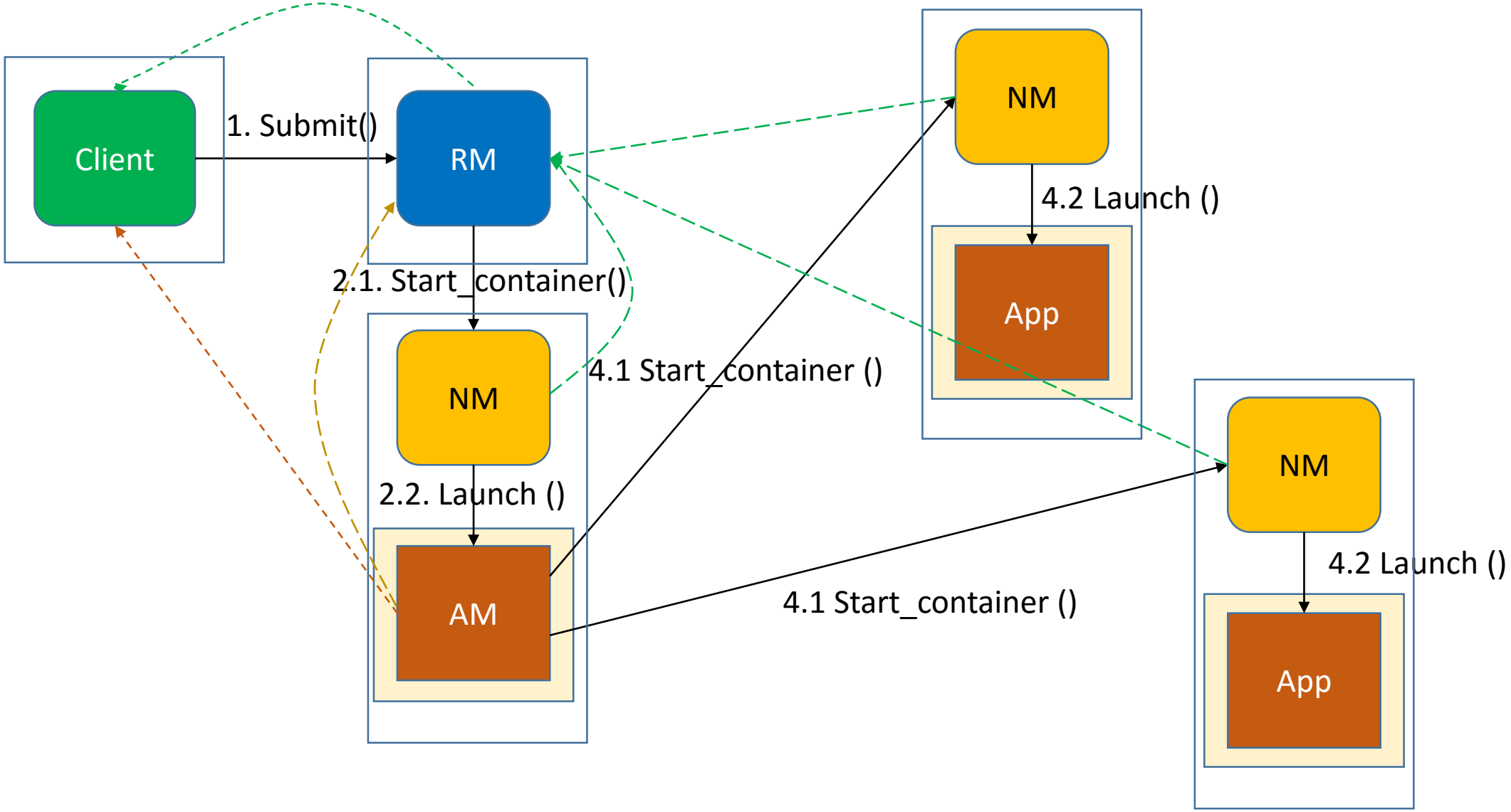
Запуск приложения



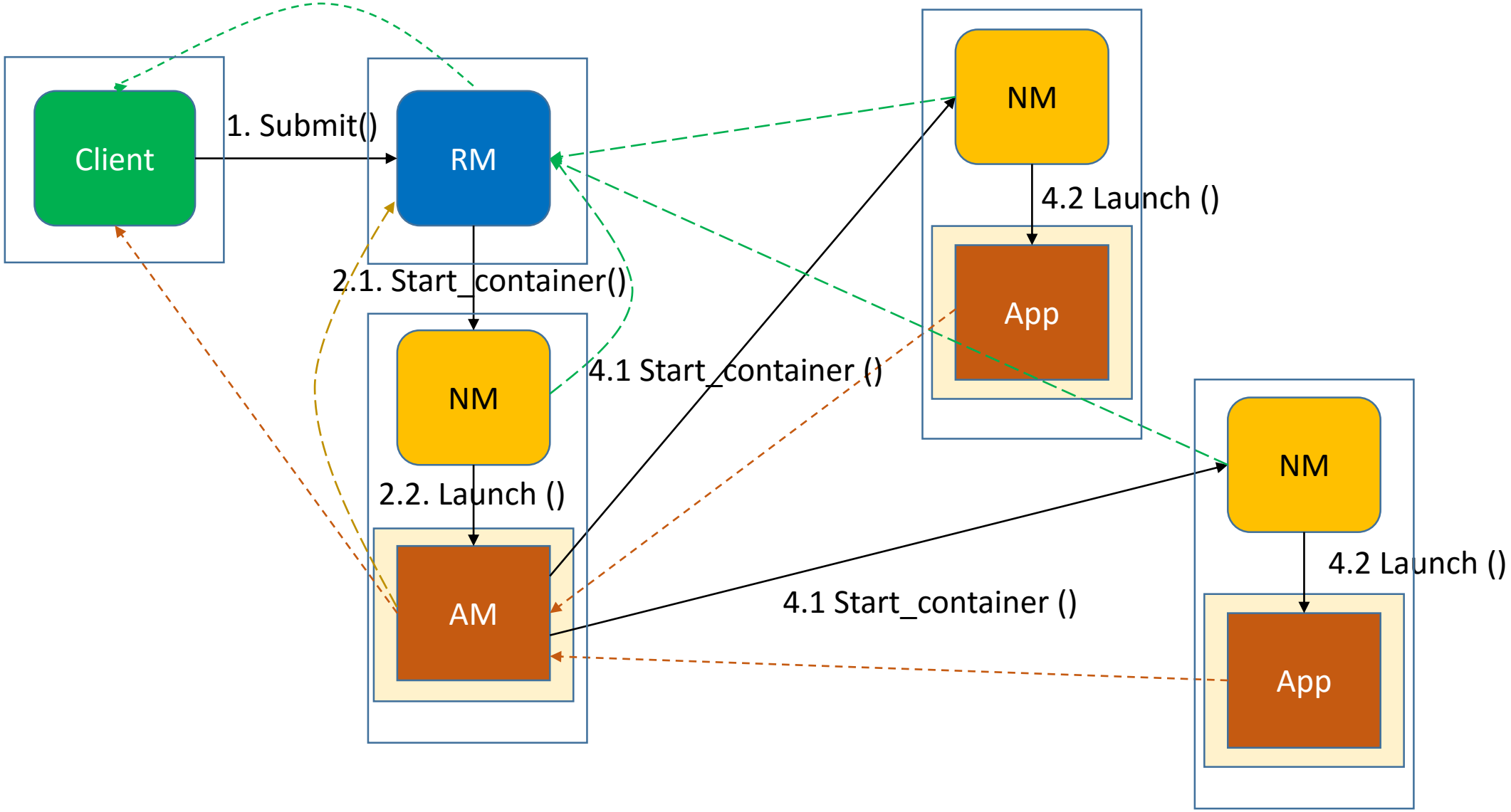
Запуск приложения



Запуск приложения



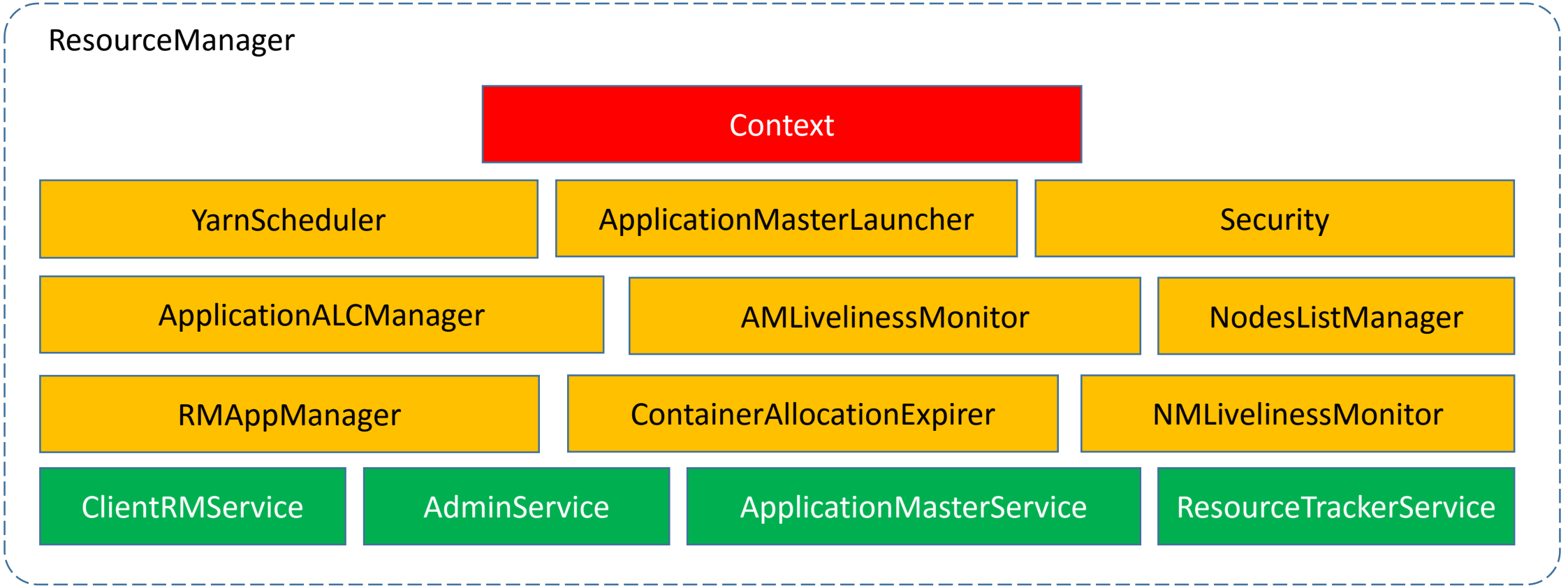
Запуск приложения



ИНСТИТУТ
ЦИФРОВОГО
ОБРАЗОВАНИЯ
МГПУ

Компоненты ResourceManager

ResourceManager

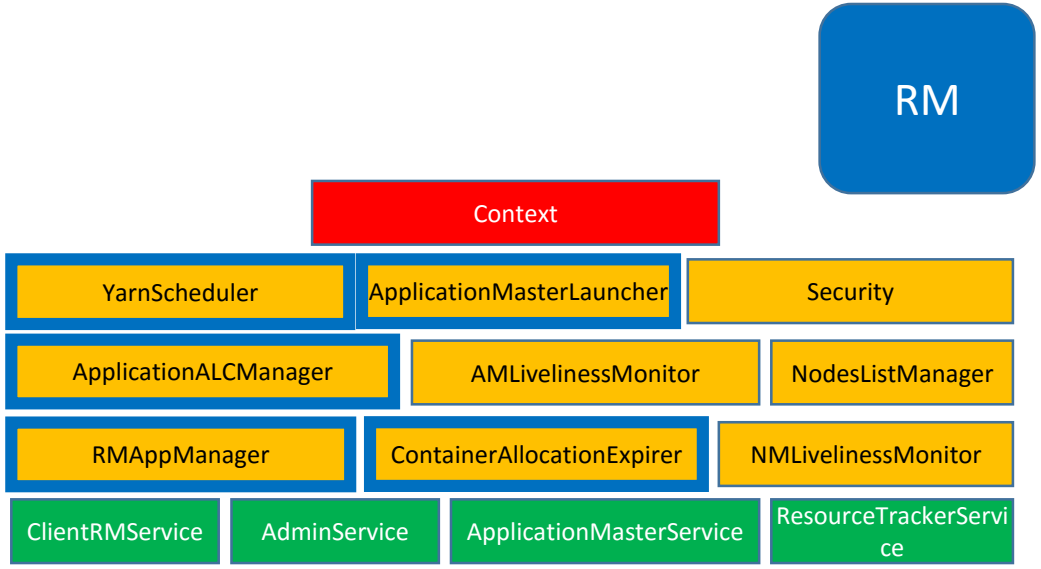


ResourceManager

RMAppManager управляет списком приложений

ApplicationACLsManager управляет ACL списком

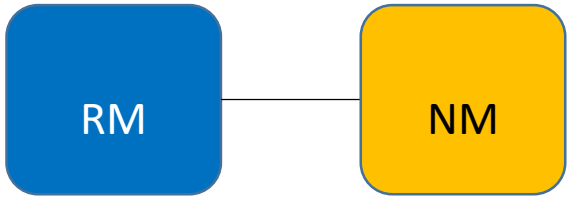
ApplicationMasterLauncher управляет набором потоков для запуска ApplicationMaster



ContainerAllocationExpirer отслеживает, чтобы использовались все выделенные container’ы

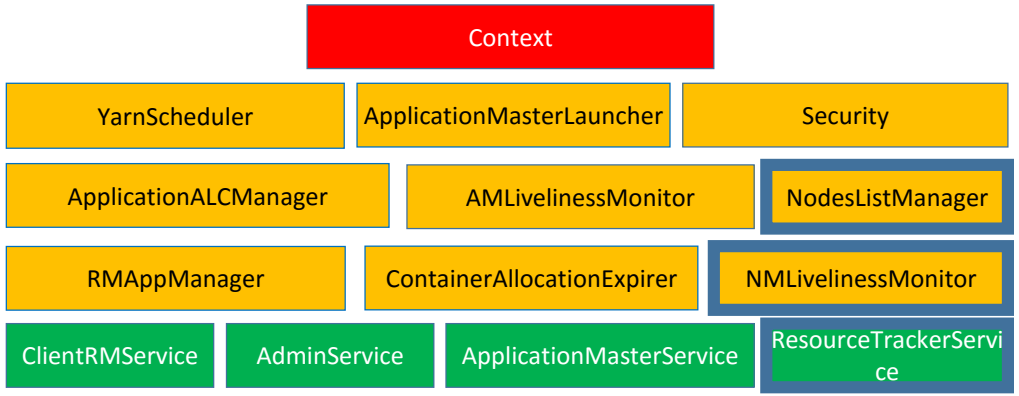
YarnScheduler интерфейс, который используется для планирования размещения ресурсов и их освобождения

ResourceManager



NMLivelinessMonitor отслеживает работоспособность NodeManager’ов

NodesListManager список выполняющих NodeManager’ов

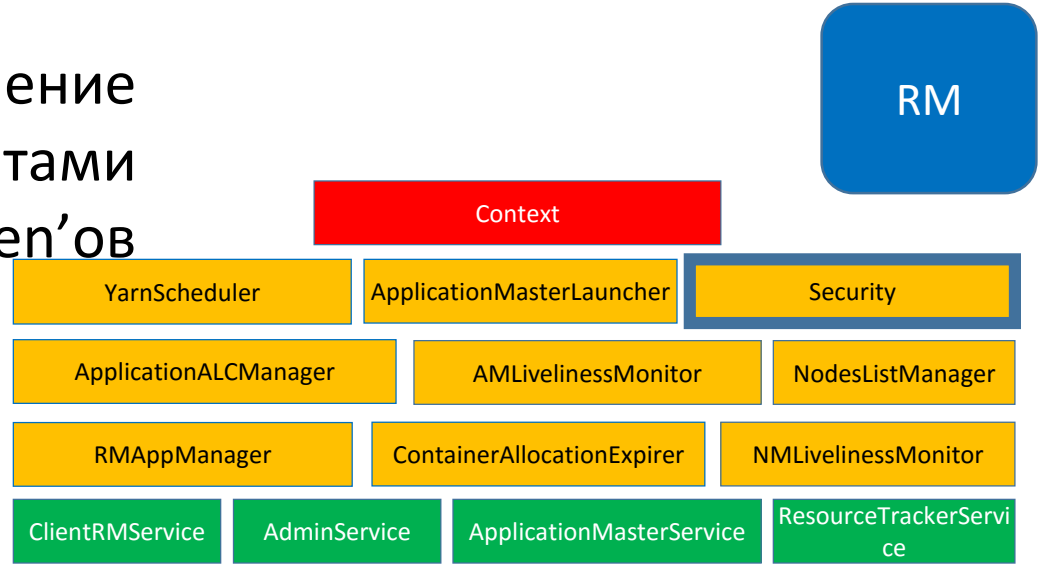


ResourceTrackerService управляет Nodemanager’ами: регистрирует новые; принимает и отвечает на heartbeat; обрабатывает статусы container’ов на Nodemanager’ах

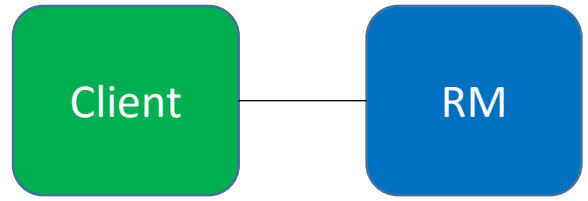
ИНСТИТУТ ЦИФРОВОГО ОБРАЗОВАНИЯ МГПУ

ResourceManager

Набор компонентов, отвечающих за обеспечение безопасного взаимодействия между компонентами YARN за счет использования token’ов (авторизация/аутентификация запросов)

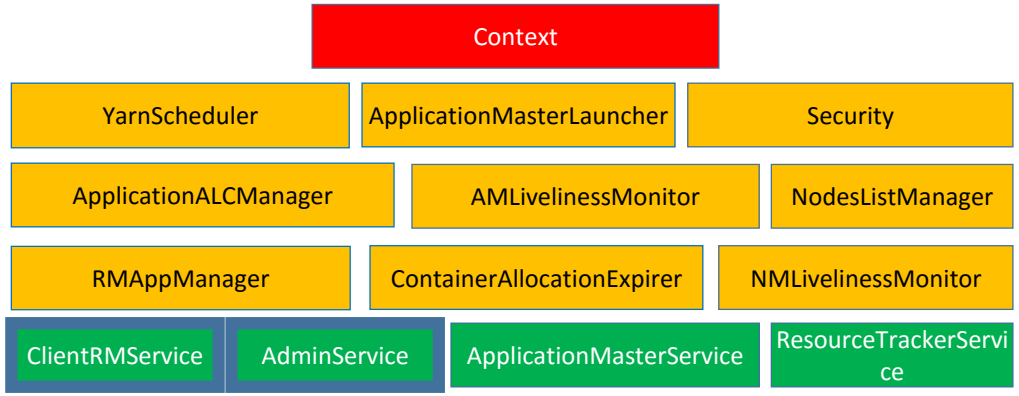


ResourceManager



ClientRMService отвечает за взаимодействие с клиентов:

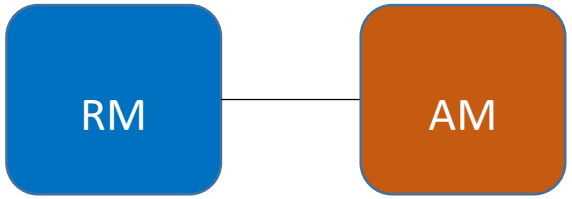
- Команду на запуск приложения
- Проверка доступа
- Отчет о выполнении приложений, container’ов



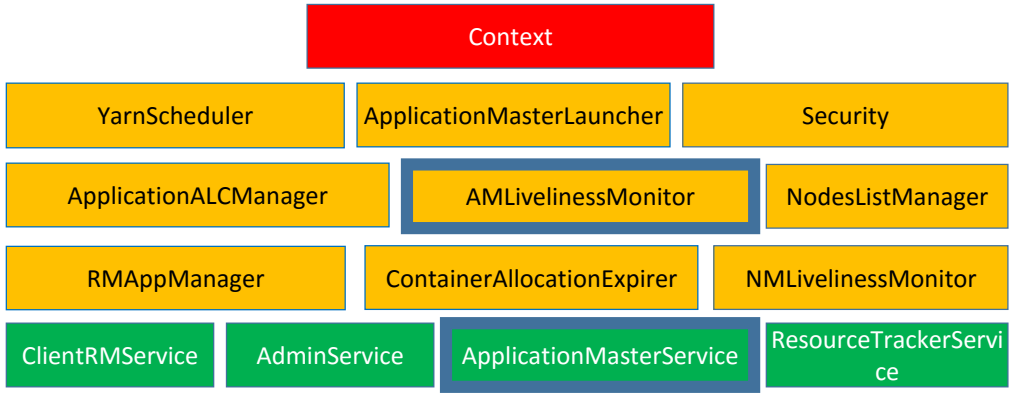
AdminService сервис администрирования. Предоставляет возможности по обновлению списков NM’ов, очередей, проверки состояния RM, ACL

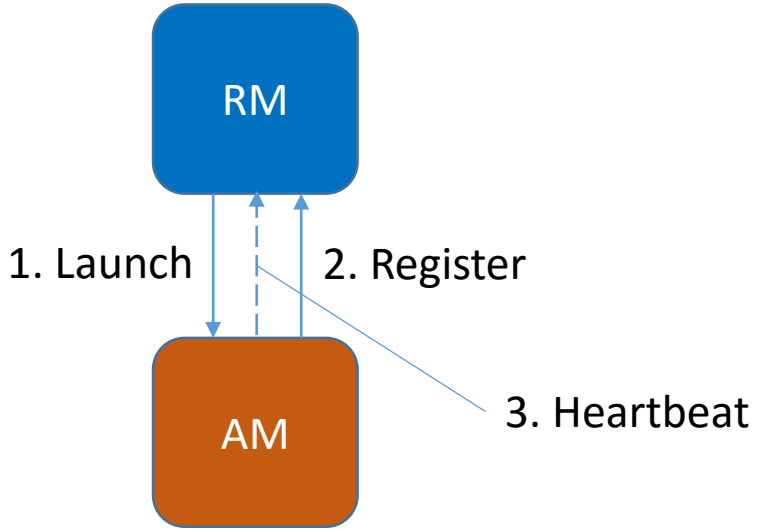
ResourceManager

ApplicationMasterService регистрирует новые AM, получает запросы на получение containerov, завершает выполнение AM



AMLivelinessMonitor отслеживает активные AM



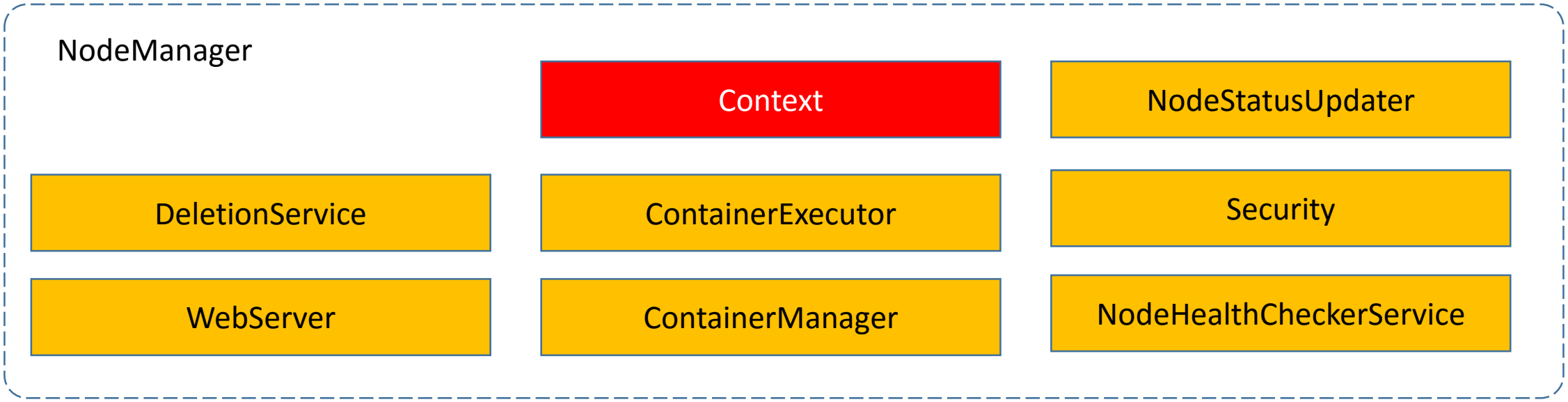


ИНСТИТУТ
ЦИФРОВОГО
ОБРАЗОВАНИЯ
МГПУ

Компоненты NodeManager

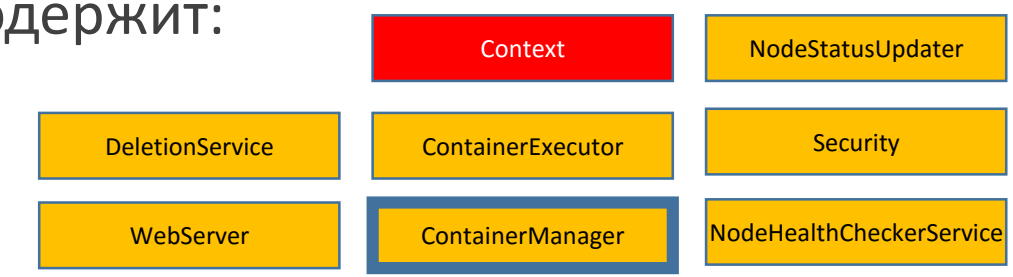
ИНСТИТУТ ЦИФРОВОГО ОБРАЗОВАНИЯ МГПУ

NodeManager



NodeManager

ContainerManager управляет container’ами на NM, содержит:

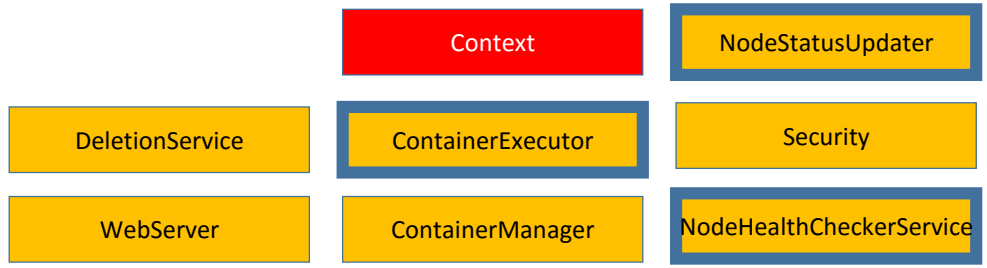


- **Server** для установки соединения
- **ContainersLauncher** служба запуска контейнеров
- **ContainersMonitorImpl** отслеживает использование ресурсов контейнером
- **ResourceLocalizationService** обеспечивает безопасную загрузку необходимых файлов для контейнера
- **AuxServices** для добавления пользовательских сервисов NM

ИНСТИТУТ ЦИФРОВОГО ОБРАЗОВАНИЯ МГПУ

NodeManager

ContainerExecutor загружает все необходимые файлы, подготавливает среду выполнения, запускает контейнер

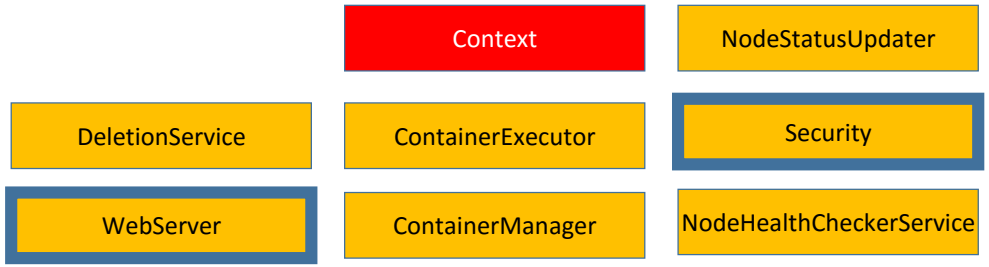


NodeHealthCheckerService обеспечивает проверку состояния NM, отправляет отчет службе, которая его запросила

NodeStatusUpdaterImpl уведомляет RM об изменении состояний контейнеров, отслеживает состояние приложений

NodeManager

WebServer предоставляет информацию о приложениях, контейнерах, узлах



Безопасность:

- **ApplicationACLsManager** обеспечивает доступ к состоянию приложения по ACL
- **RMSecretManagerService** проверяет входящие запросы авторизованы RM

ИНСТИТУТ
ЦИФРОВОГО
ОБРАЗОВАНИЯ
МГПУ

Отказоустойчивость

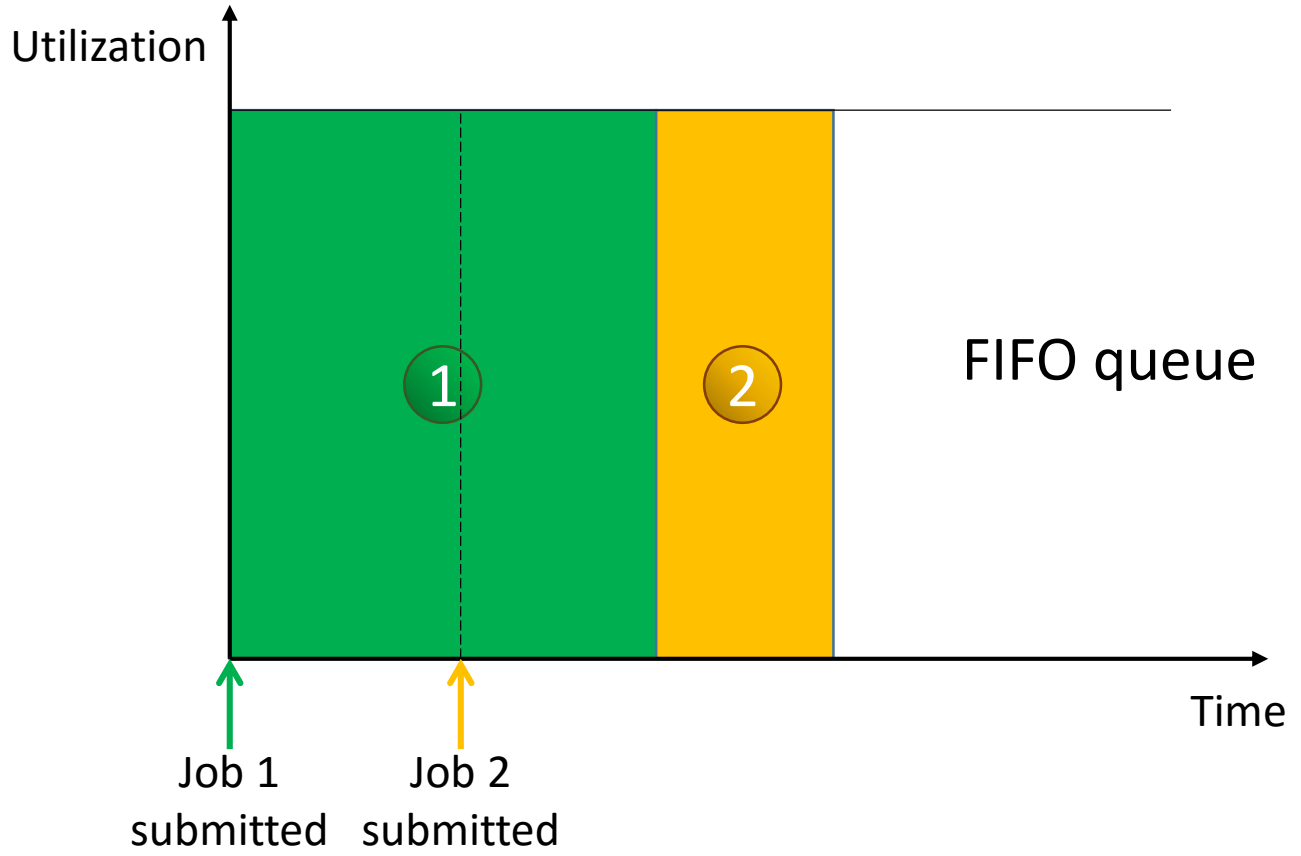
Отказоустойчивость

См. лекцию по MapReduce

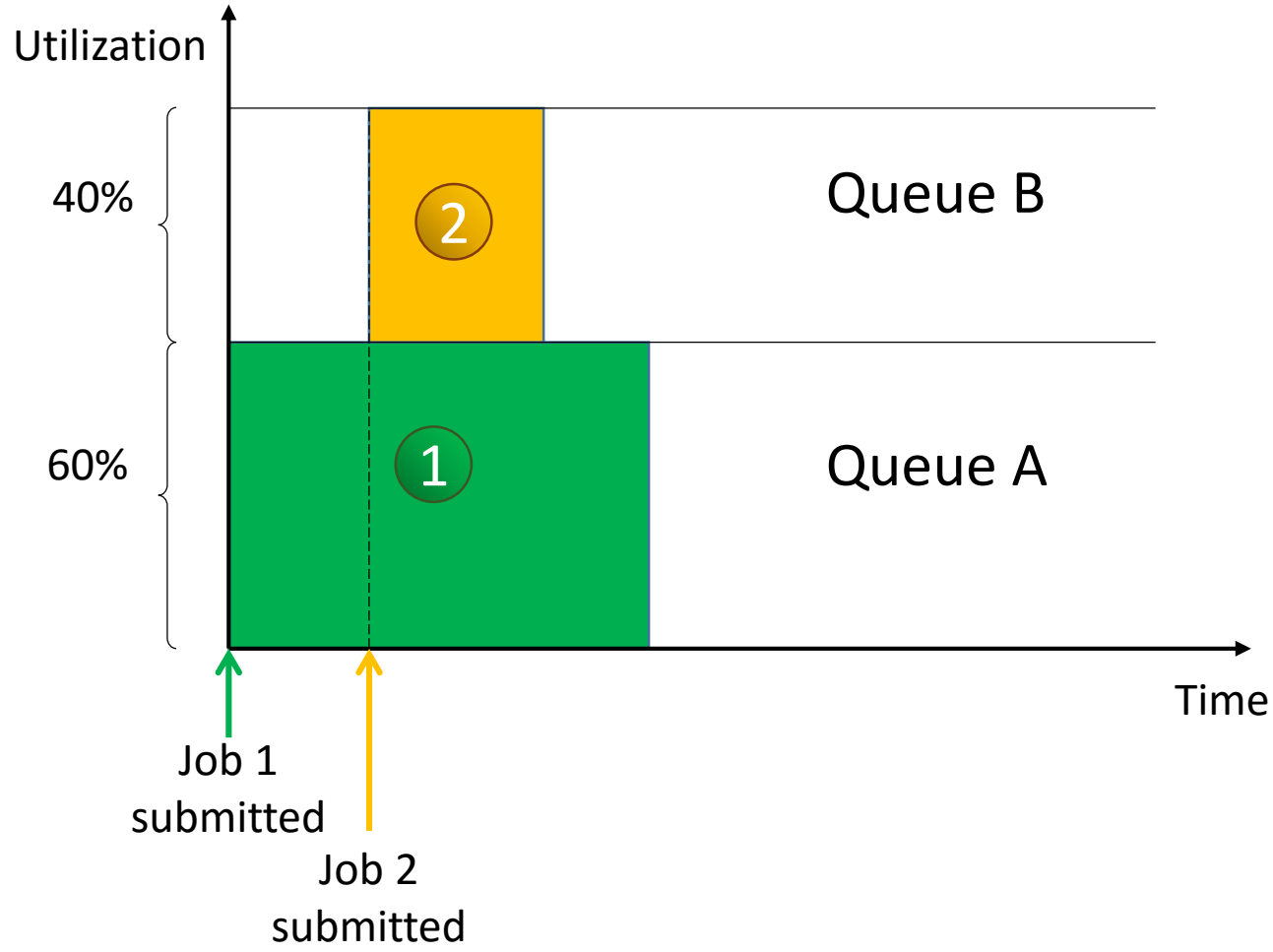
ИНСТИТУТ
ЦИФРОВОГО
ОБРАЗОВАНИЯ
МГПУ

Планирование выполнения (Scheduling)

FIFO


Utilization
1
2
FIFO queue
Time
Job 1 submitted
Job 2 submitted

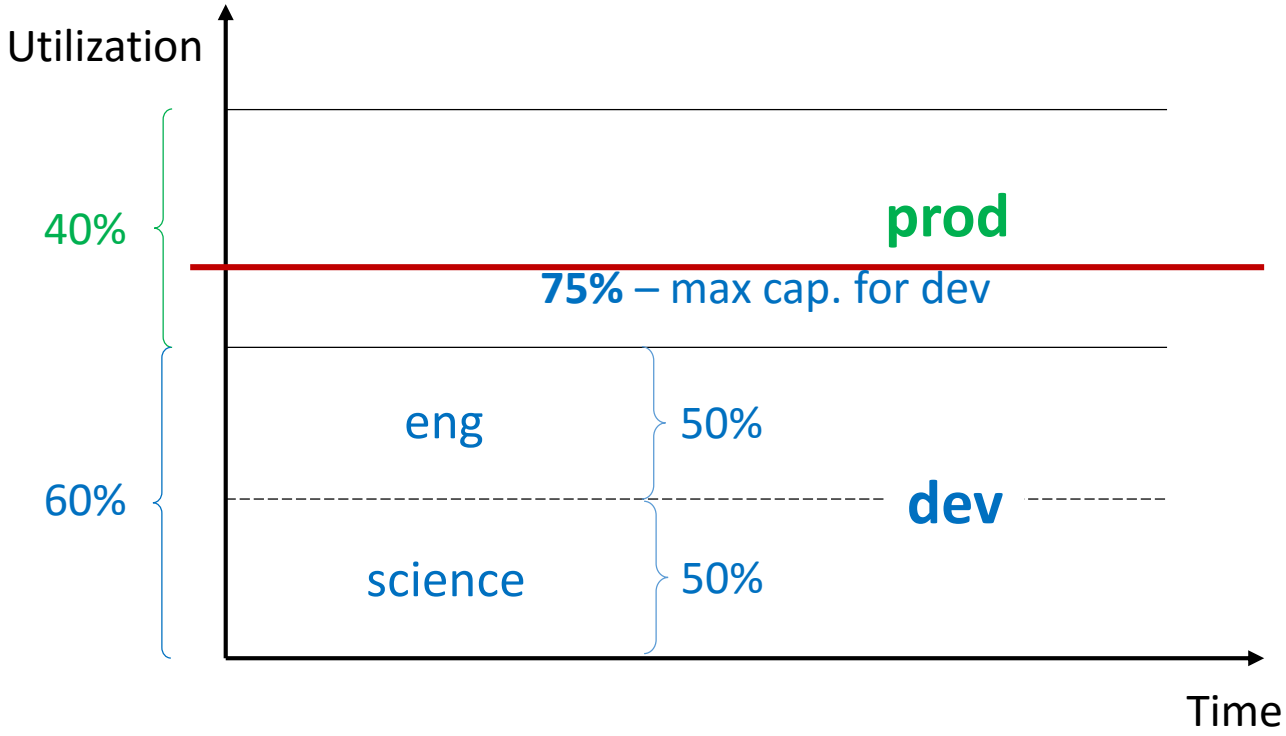
Capacity Scheduler



Capacity Scheduler

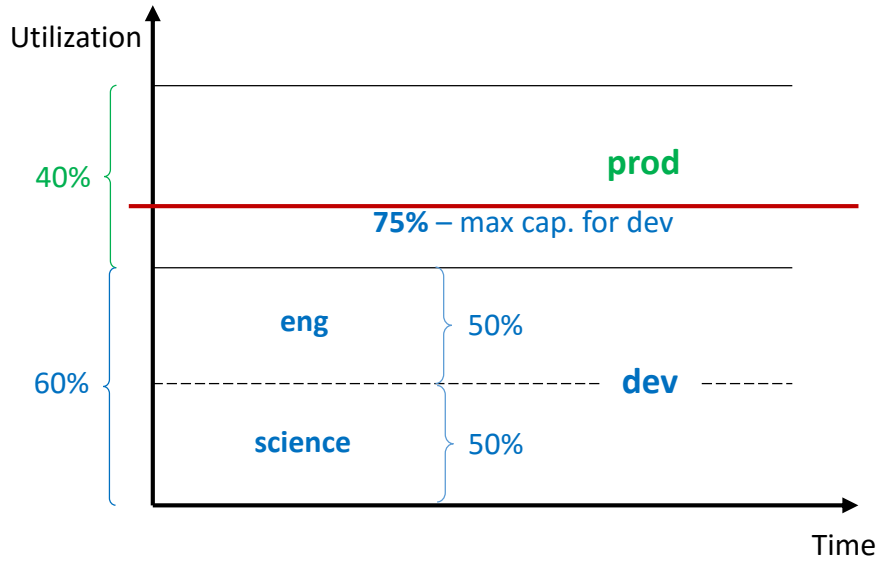
Queue hierarchy

- root
  - prod – 40%
  - dev – 60%
    - eng – 50%
    - science – 50%



Capacity Scheduler

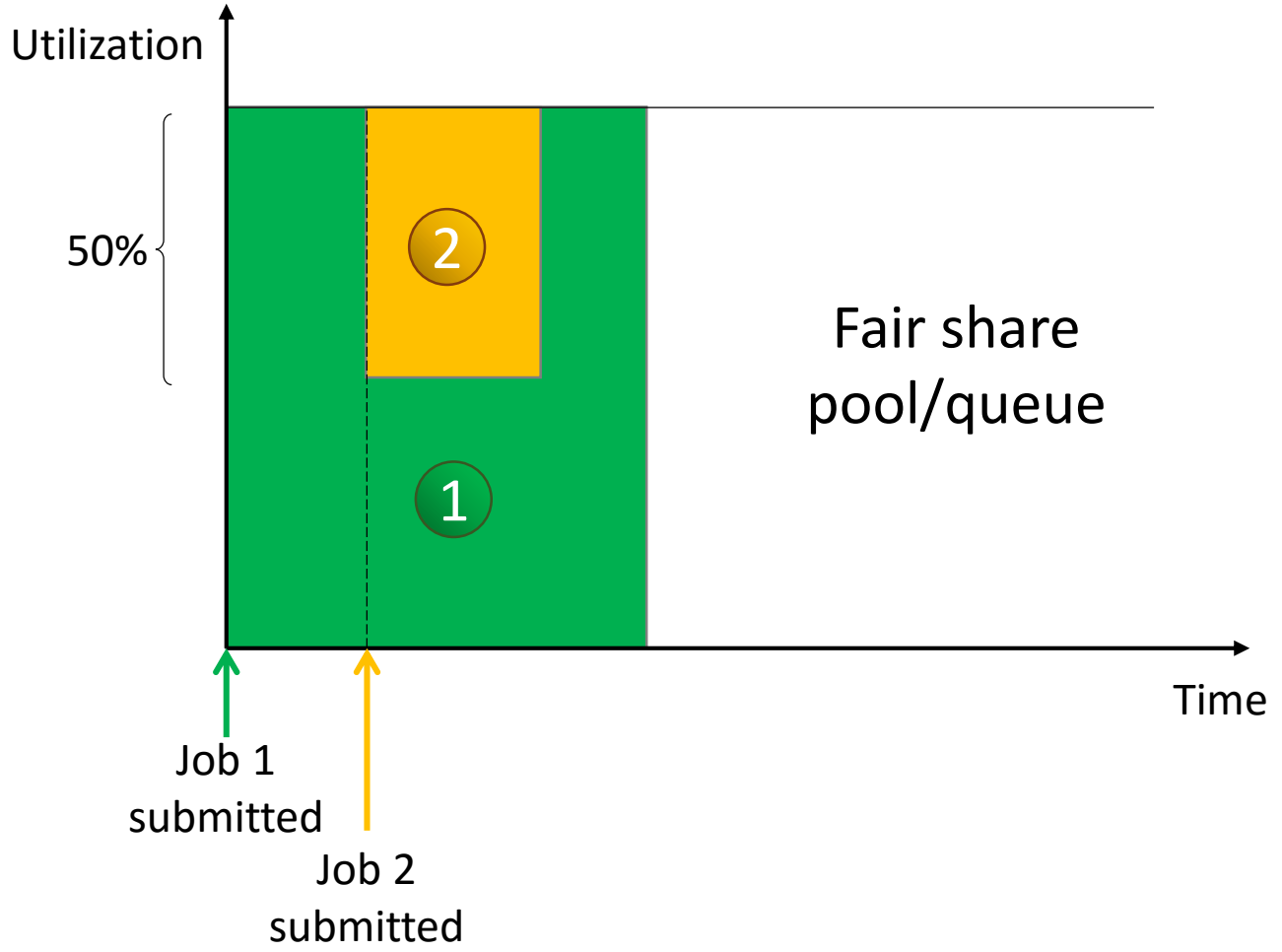
```
<?xml version="1.0"?>
<configuration>
  <property>
    <name>yarn.scheduler.capacity.root.queues</name>
    <value>prod,dev</value>
  </property>
  <property>
    <name>yarn.scheduler.capacity.root.dev.queues</name>
    <value>eng,science</value>
  </property>
  <property>
    <name>yarn.scheduler.capacity.root.prod.capacity</name>
    <value>40</value>
  </property>
  <property>
    <name>yarn.scheduler.capacity.root.dev.capacity</name>
    <value>60</value>
  </property>
  <property>
    <name>yarn.scheduler.capacity.root.dev.maximum-capacity</name>
    <value>75</value>
  </property>
  <property>
    <name>yarn.scheduler.capacity.root.dev.eng.capacity</name>
    <value>50</value>
  </property>
  <property>
    <name>yarn.scheduler.capacity.root.dev.science.capacity</name>
    <value>50</value>
  </property>
</configuration>
```



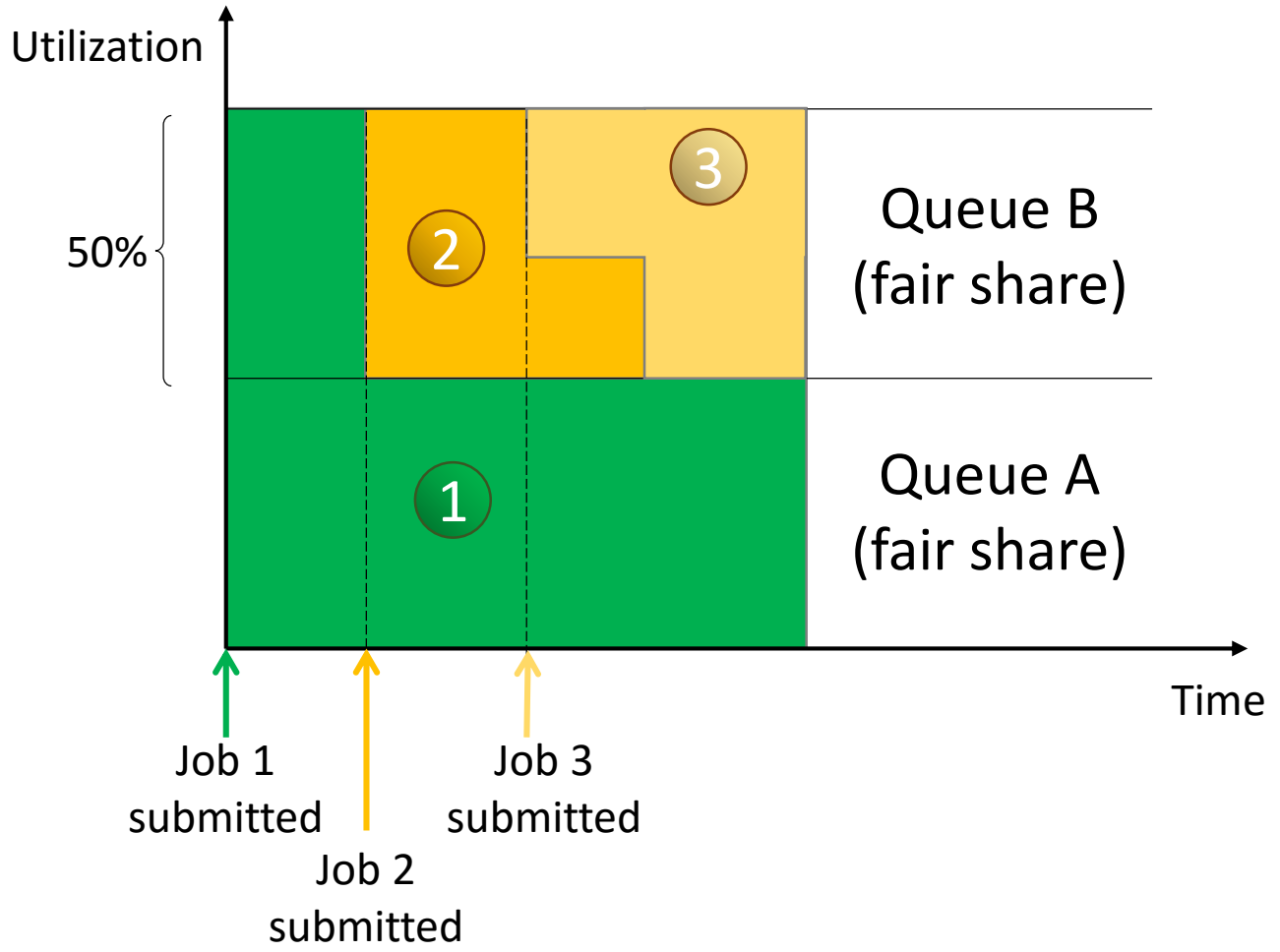
capacity-scheduler.xml

yarn-site.xml

FairScheduler



FairScheduler



FairScheduler

fair-scheduler.xml

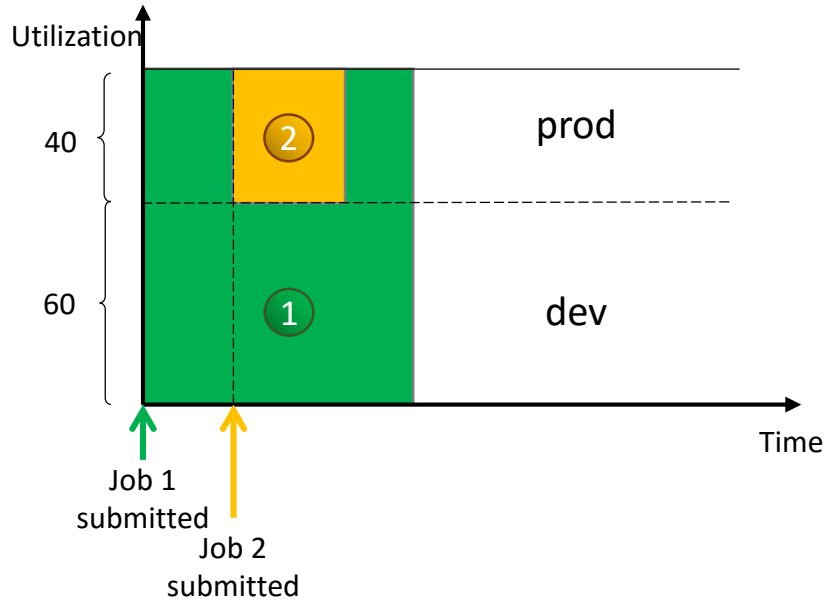
```
<?xml version="1.0"?>
<allocations>
  <defaultQueueSchedulingPolicy>fair</defaultQueueSchedulingPolicy>

  <queue name="prod">
    <weight>40</weight>
    <schedulingPolicy>fifo</schedulingPolicy>
  </queue>

  <queue name="dev">
    <weight>60</weight>
    <queue name="eng" />
    <queue name="science" />
  </queue>

  <queuePlacementPolicy>
    <rule name="specified" create="false" />
    <rule name="primaryGroup" create="false" />
    <rule name="default" queue="dev.eng" />
  </queuePlacementPolicy>
</allocations>
```

yarn-site.xml



Dominant Resource Fairness

- Cluster: (100 CPUs, 10 TB)
- App A per task: (2 CPUs, 300 GB)
- App B per task: (6 CPUs, 100 GB)

A

$$\frac{2\ CPUs}{100\ CPUs} = 0.02 \qquad \frac{300\ GB}{10\ TB} = 0.03$$

B

$$\frac{6\ CPUs}{100\ CPUs} = 0.06 \qquad \frac{100\ GB}{10\ TB} = 0.01$$

$$0.03 \cdot x = 0.06 \cdot y \Rightarrow \frac{x}{y} = 2 \Rightarrow$$

App A – 20 tasks

App B – 10 tasks

ИНСТИТУТ ЦИФРОВОГО ОБРАЗОВАНИЯ МГПУ

Источники

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

Hadoop (github source code)

Apache Hadoop YARN (doc)

Tuning YARN (blog)

Managing CPU Resources in your Hadoop YARN Clusters (blog)

Apache Hadoop YARN in HDP 2.2: Isolation of CPU resources in your Hadoop YARN clusters (blog)

YARN Container Launch Details (blog)

Apache Hadoop YARN – ResourceManager (blog)

Apache Hadoop YARN – NodeManager (blog)

Best Practices for YARN Resource Management (blog)