YTsaurus

Пишем свой cluster manager для Apache Spark

Александр Токарев,
Яндекс, TeamLead группы разработки
SPYT powered by Apache Spark

08.09.2024

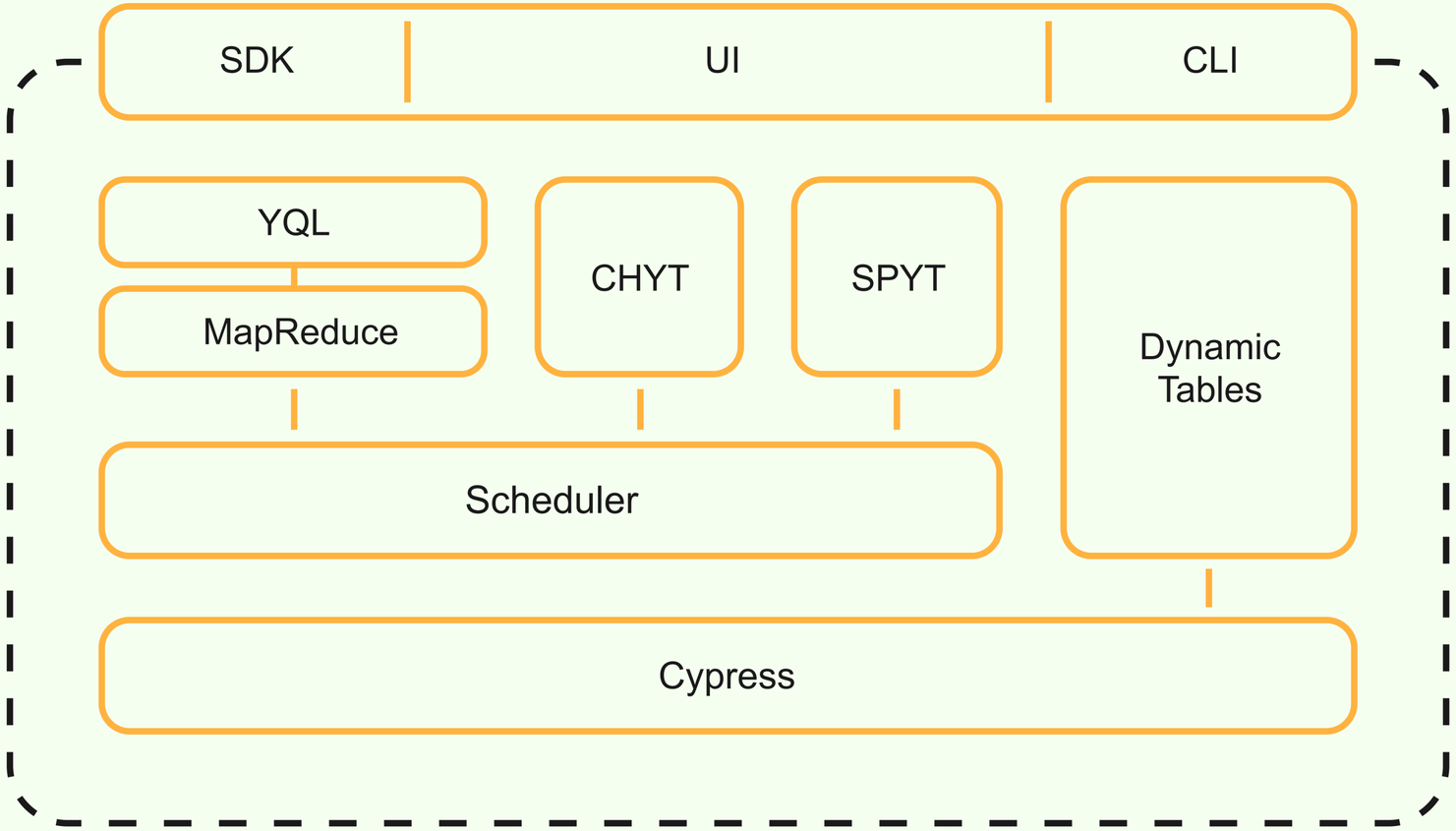
Обо мне

- Разрабатываю ПО более 15 лет
- Работаю с Apache Spark с 2015 года (с версии 1.2.0)
- Раньше — в Qiwi, CleverData, Leroy Merlin
- Сейчас — в Яндексе, тимлид проекта YTsaurus SPYT, подключаю Apache Spark к YTsaurus

Что такое YTsaurus?

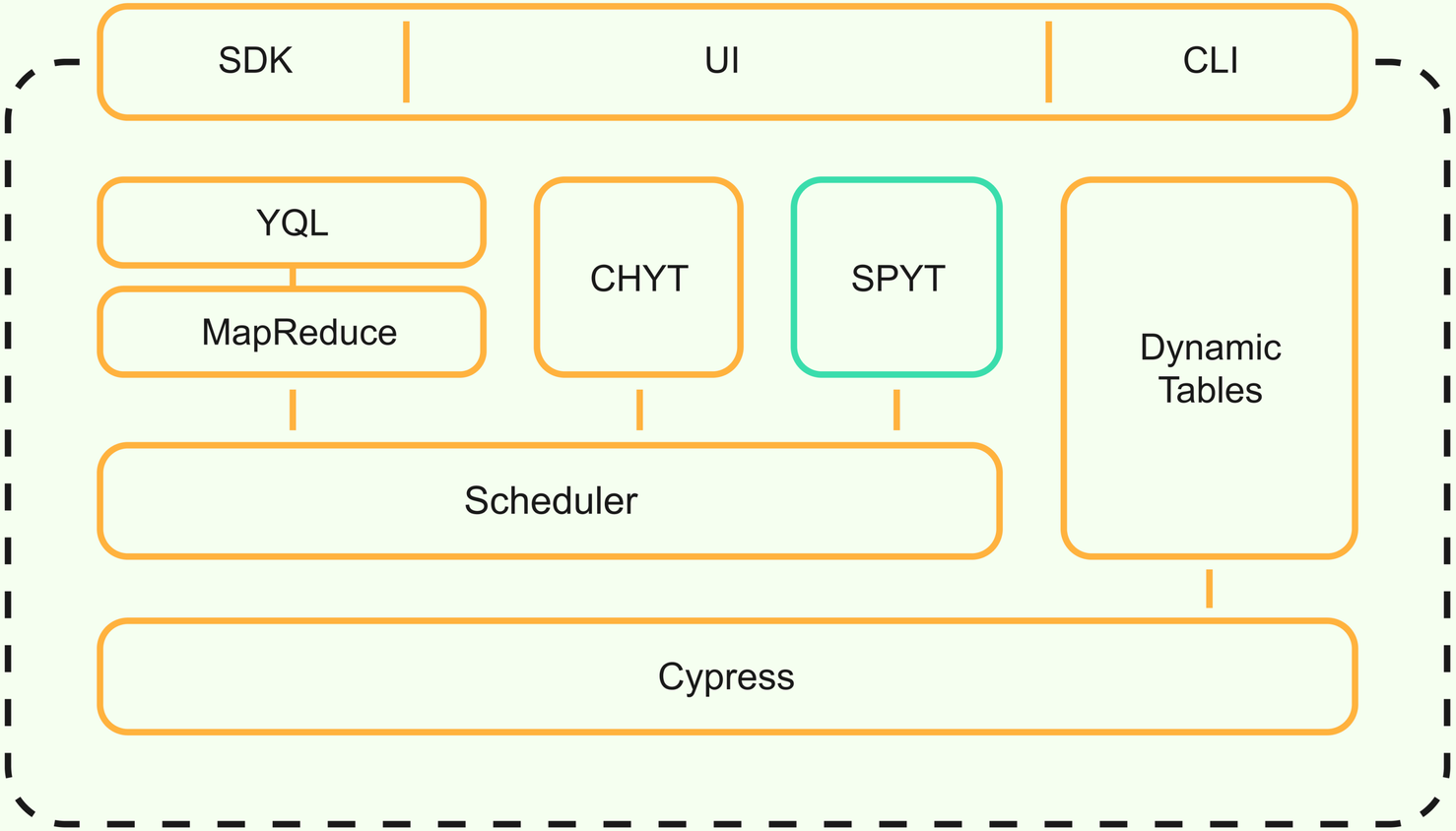
- Платформа для распределённого хранения и обработки больших данных
- Ближайший аналог — экосистема Apache Hadoop
- Собственная разработка, C++
- Создана в Яндексе
- Активно разрабатывается и используется с 2010 года
- Open Source с марта 2023
- Лицензия Apache 2.0

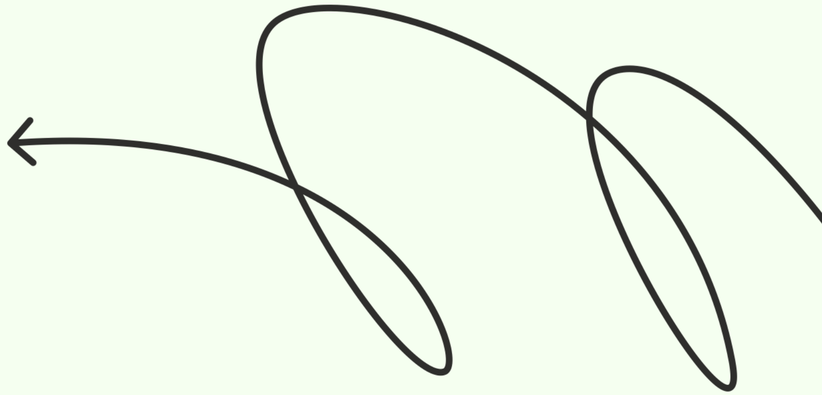
Основные компоненты YTsaurus

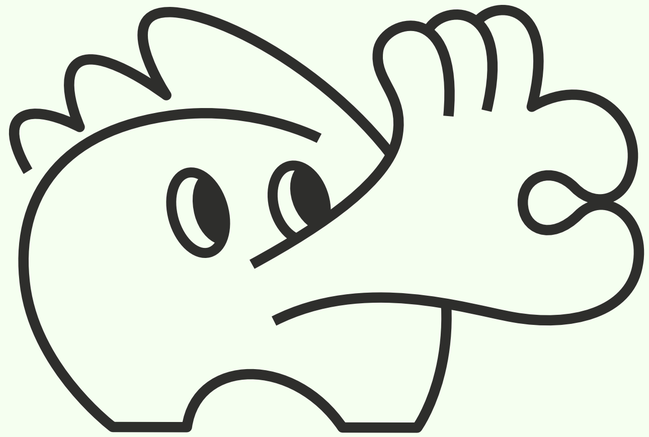


YTsaurus	Hadoop ecosystem
Cypress	HDFS + ZooKeeper + Hive Metastore
Dynamic Tables	HBase
YTsaurus MapReduce	Hadoop MapReduce
YQL	HiveQL
CHYT	Presto/Trino/ Impala
SPYT	Apache Spark

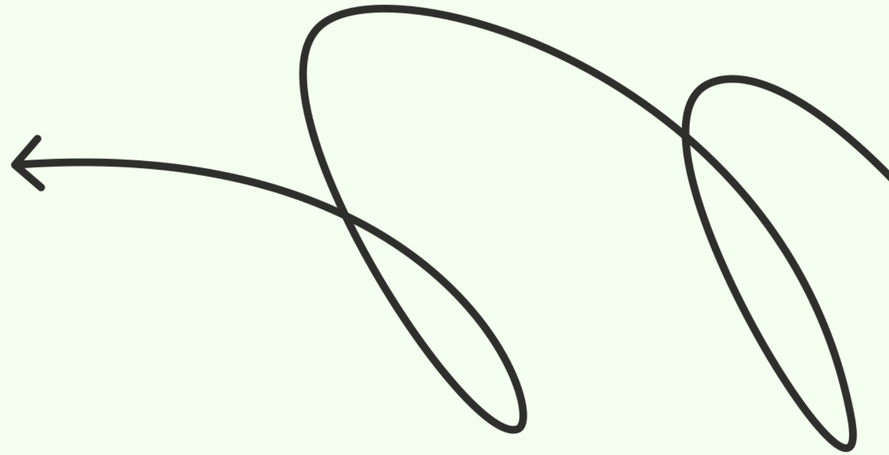
SPYT powered by Apache Spark





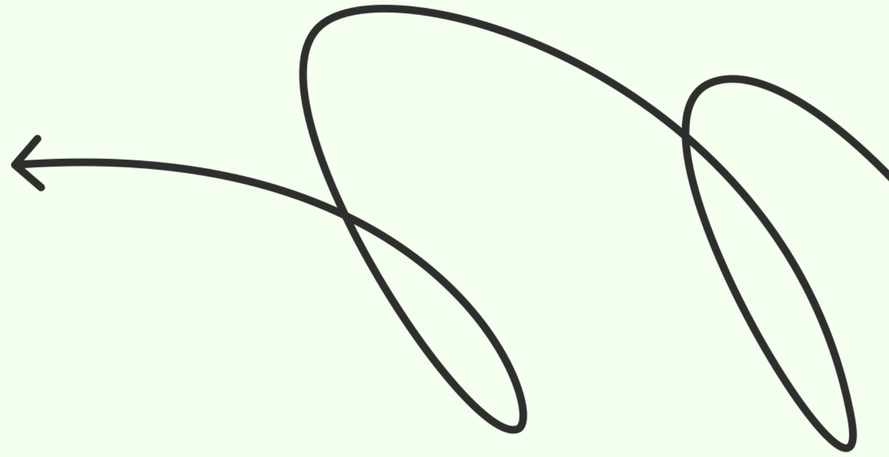


Фреймворк для выполнения вычислений



Фреймворк для выполнения вычислений

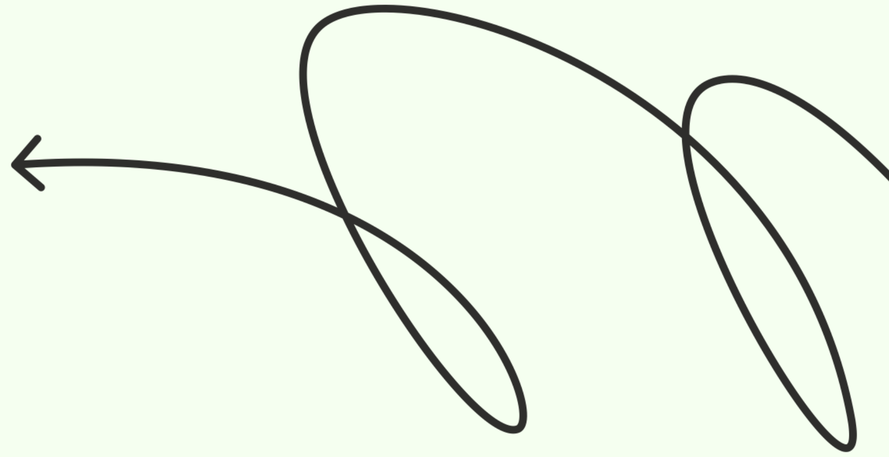
- над большими объёмами



APACHE Spark™

Фреймворк для выполнения вычислений

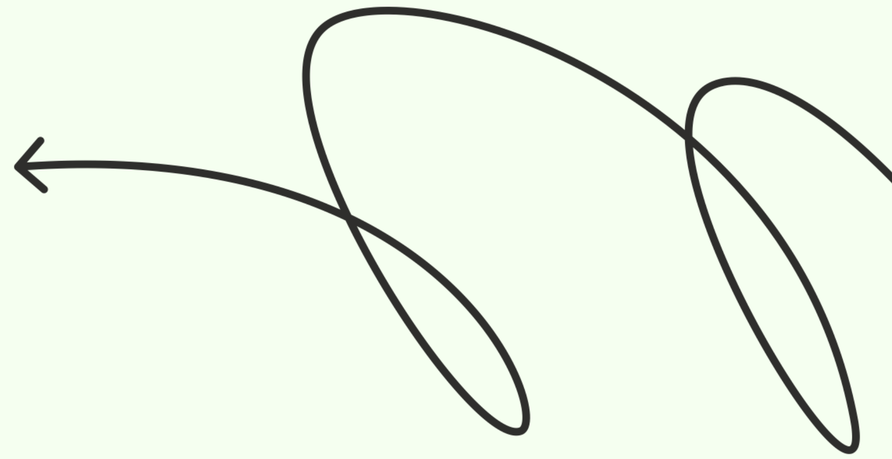
- над большими объёмами
- распределёнными



APACHE Spark™

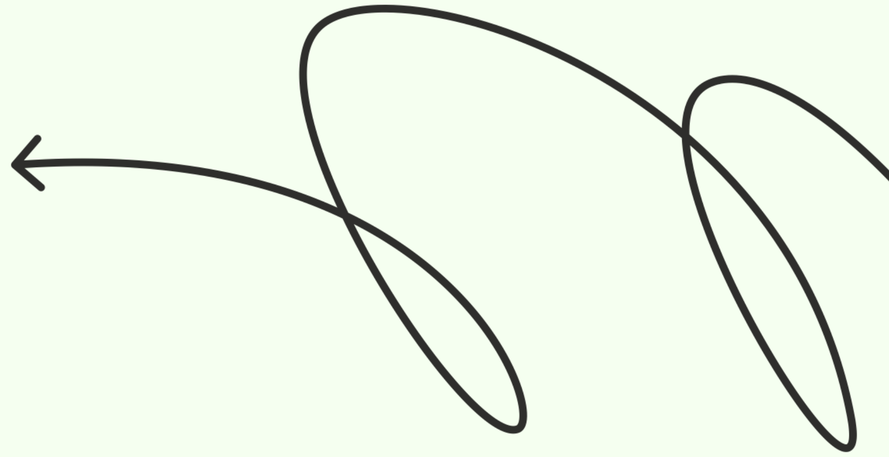
Фреймворк для выполнения вычислений

- над большими объёмами
- распределёнными
- не всегда структурированных



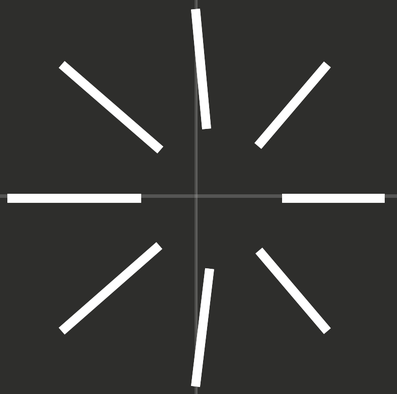
Фреймворк для выполнения вычислений

- над большими объёмами
- распределёнными
- не всегда структурированных
- данных



Apache Spark™

Для этого ему необходимо уметь запускать распределённые процессы, которые логически связаны между собой

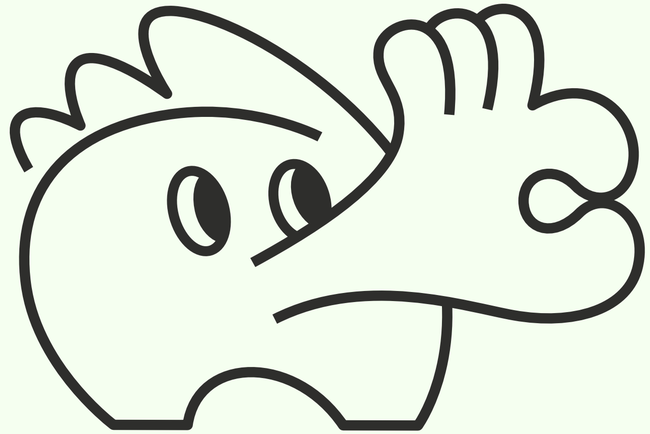


Spark cluster managers

Назначение cluster manager

Основная задача cluster manager —

выделение вычислительных ресурсов для выполнения джобы в соответствии с требуемым запросом (CPU, память)



Какие бывают кластер-менеджеры?

Встроенные

Реализация в spark core,
local и standalone

Какие бывают кластер-менеджеры?

Встроенные

Реализация в spark core, local и standalone

Подключаемые

Все остальные

Local

- Запуск всех компонентов внутри одного процесса

Local

- Запуск всех компонентов внутри одного процесса
- Подходит для разработки, отладки и тестирования

Local

- Запуск всех компонентов внутри одного процесса
- Подходит для разработки, отладки и тестирования
- spark-submit --master local — одно ядро

Local

- Запуск всех компонентов внутри одного процесса
- Подходит для разработки, отладки и тестирования
- spark-submit --master local — одно ядро
- spark-submit --master local[K] — K ядер

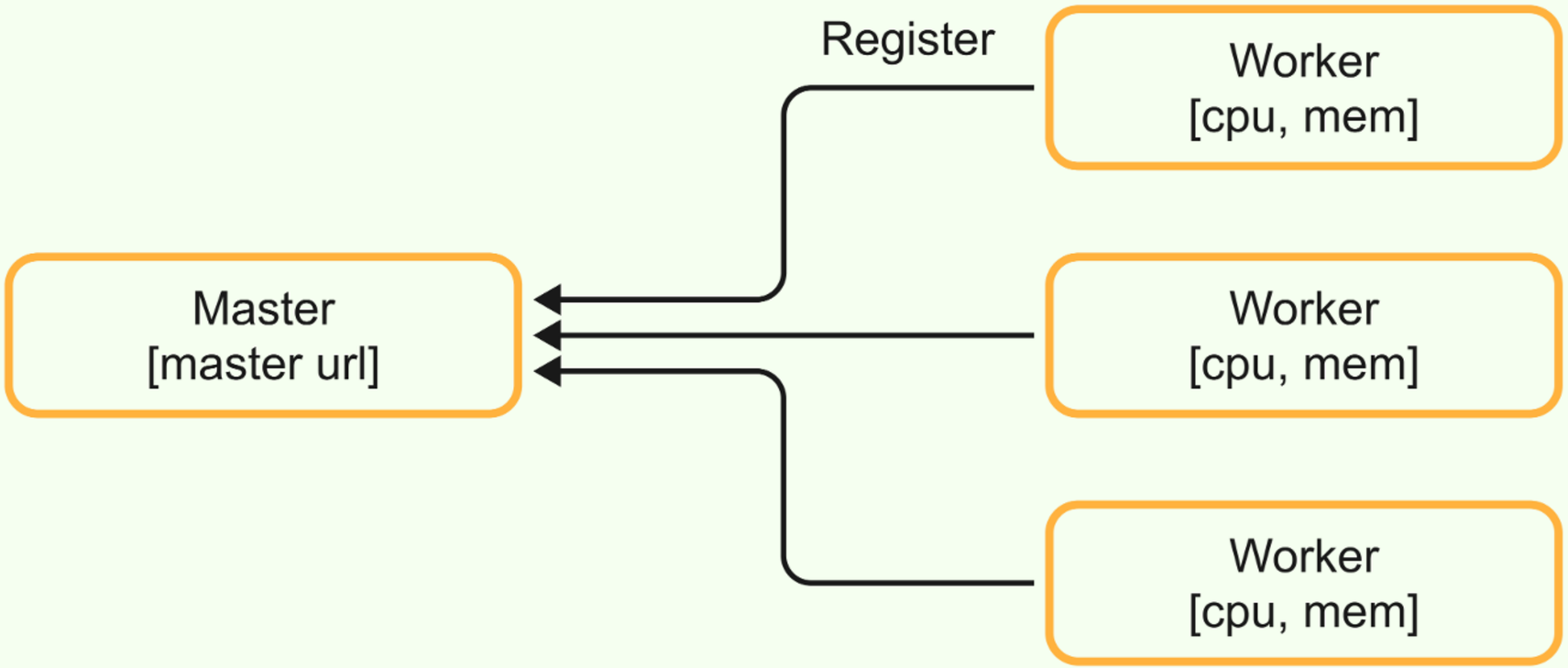
Local

- Запуск всех компонентов внутри одного процесса
- Подходит для разработки, отладки и тестирования
- spark-submit --master local — одно ядро
- spark-submit --master local[K] — K ядер
- spark-submit --master local[*] — все доступные ядра

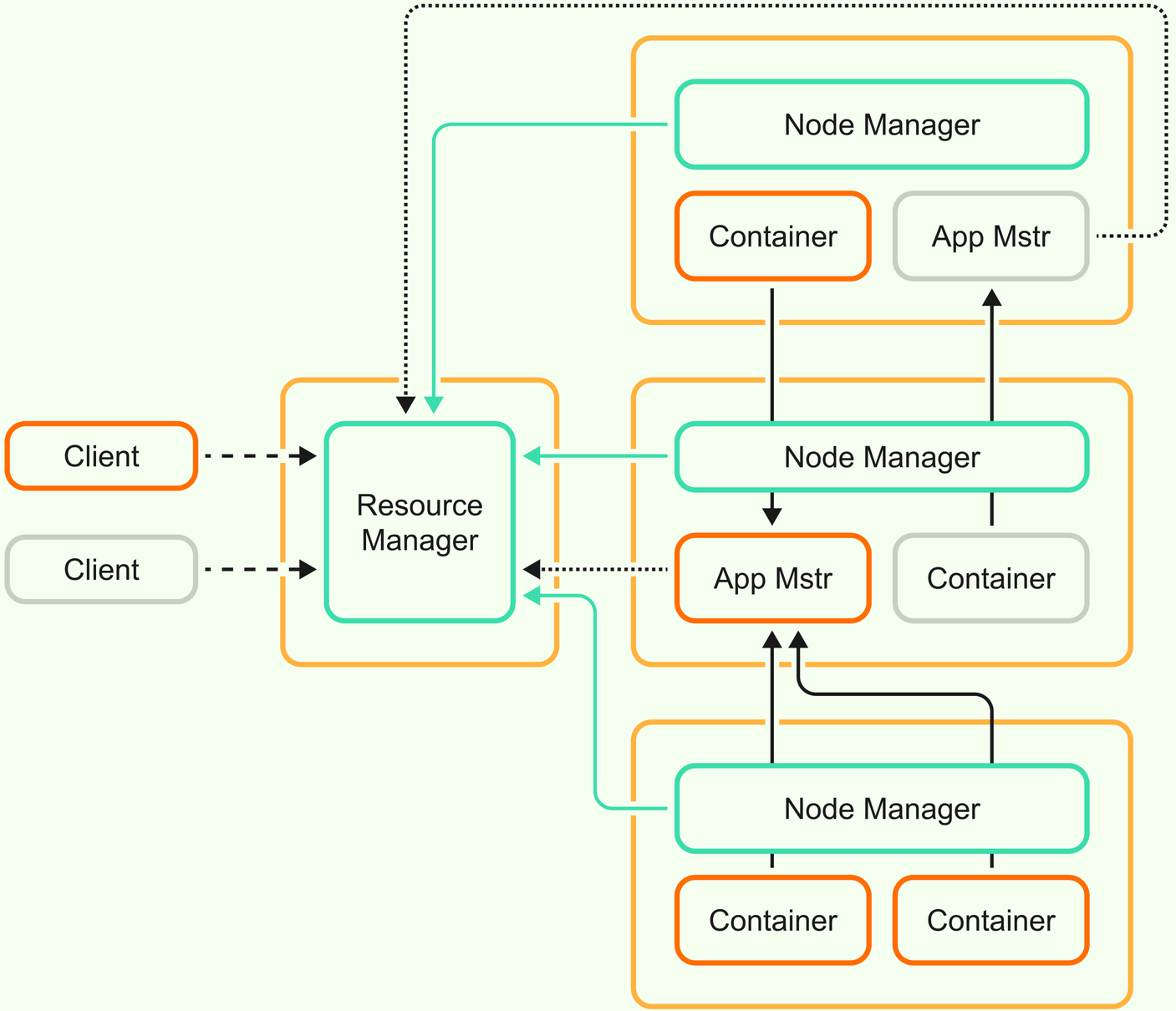
Standalone

./sbin/start-master.sh

./sbin/start-worker.sh <master-spark-URL>



В отличие от других кластер менеджеров, для YARN конфигурация поставляется в виде набора xml-файлов, вроде core-site.xml, hdfs-site.xml, yarn-site.xml и т. д.



MapReduce Status

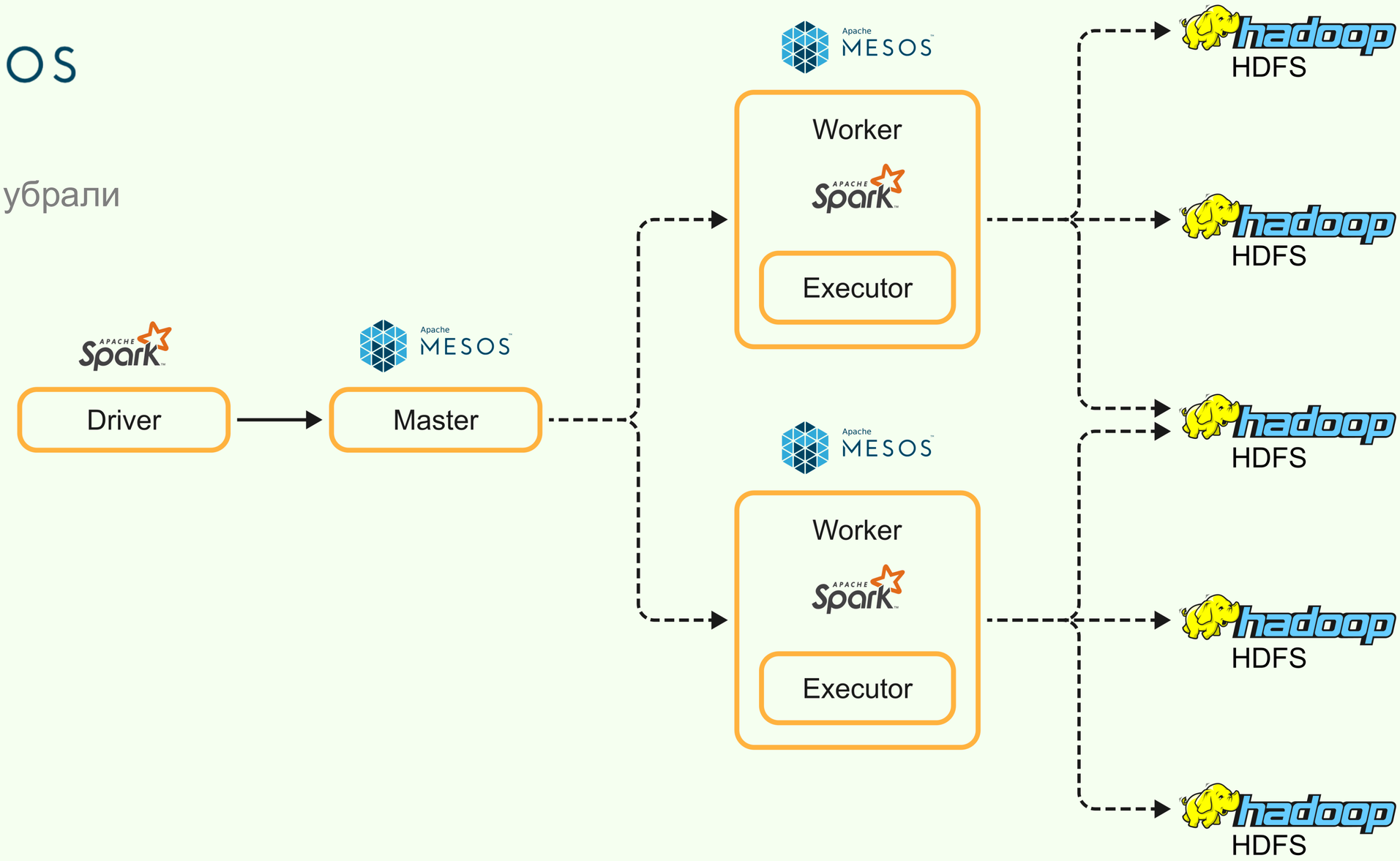
Job Submission

Node Status

Resource Request

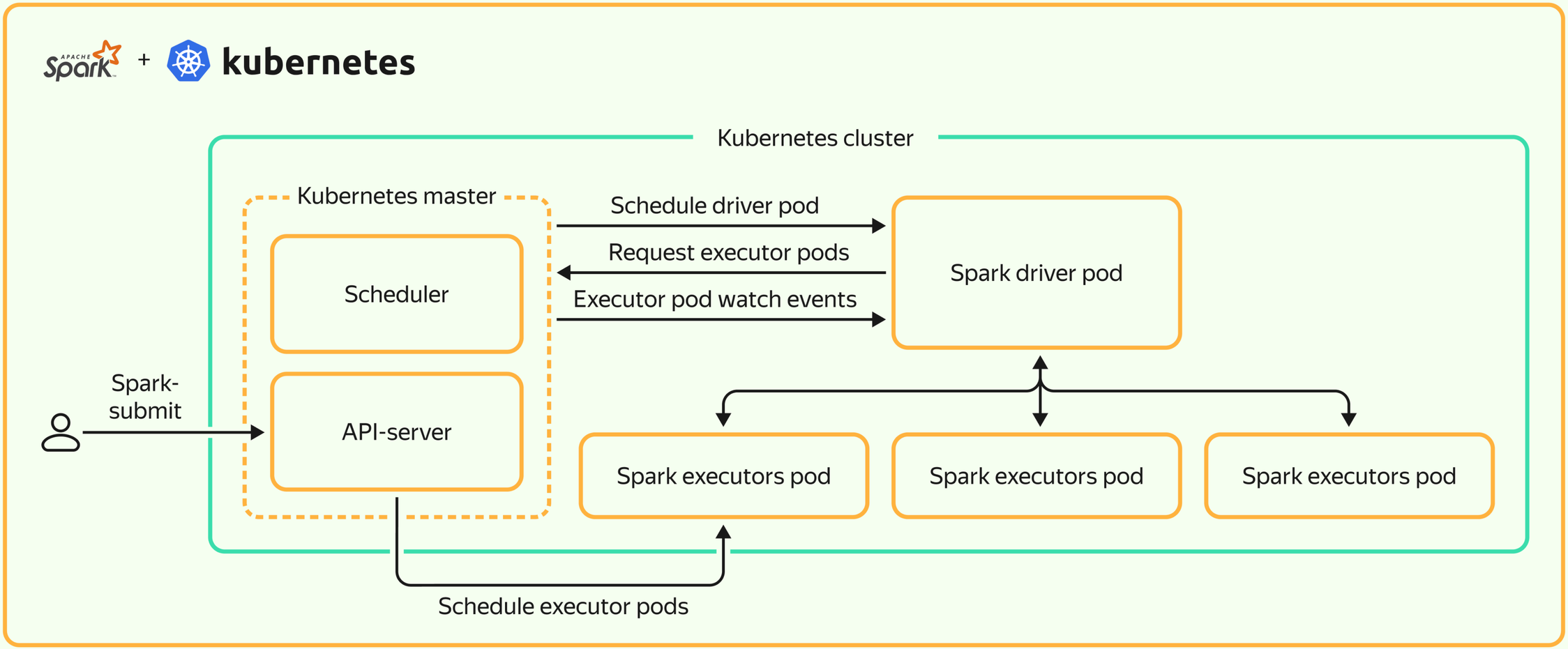
MESOS

В версии 4.0.0 убрали



kubernetes

Появился в версии 2.2.0 в виде экспериментального модуля, с 2.3.0 — прод



Кластер-менеджеры в Spark

Встроенные:

- Local
- Standalone

Кластер-менеджеры в Spark

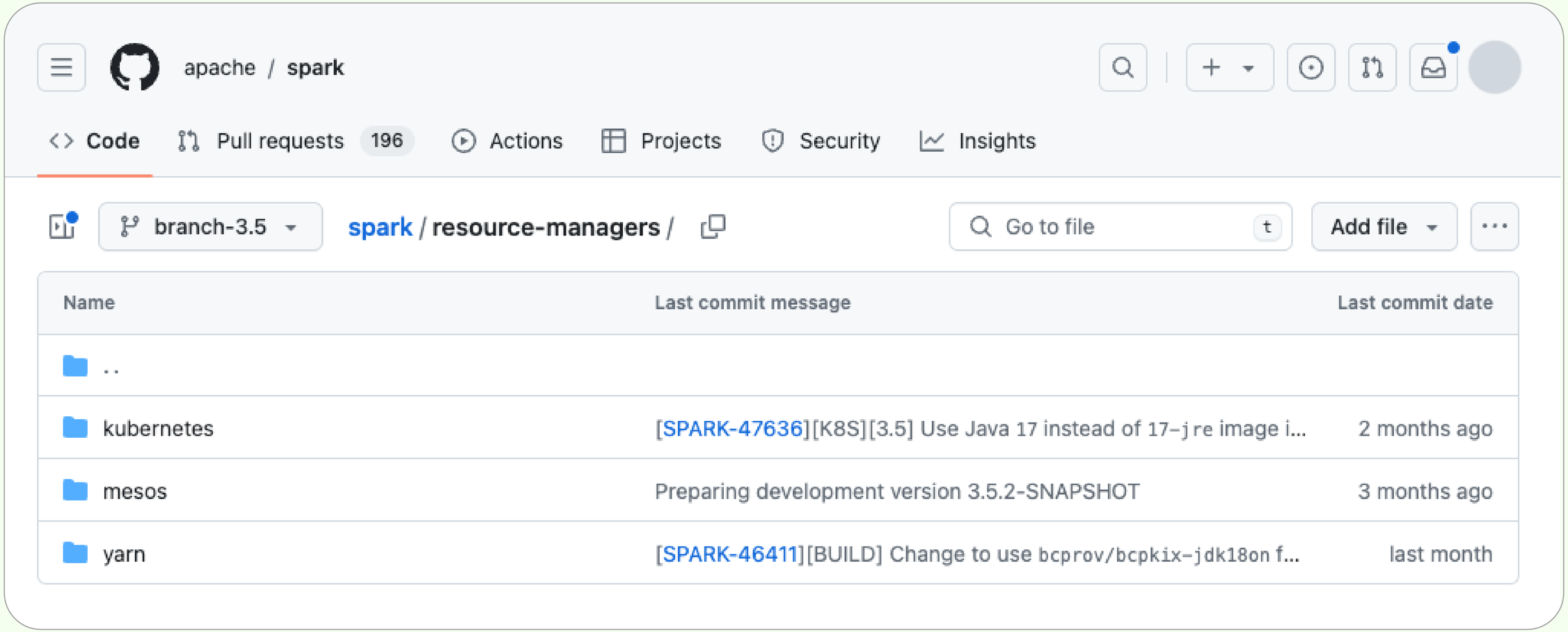
Встроенные:

- Local
- Standalone

Подключаемые:

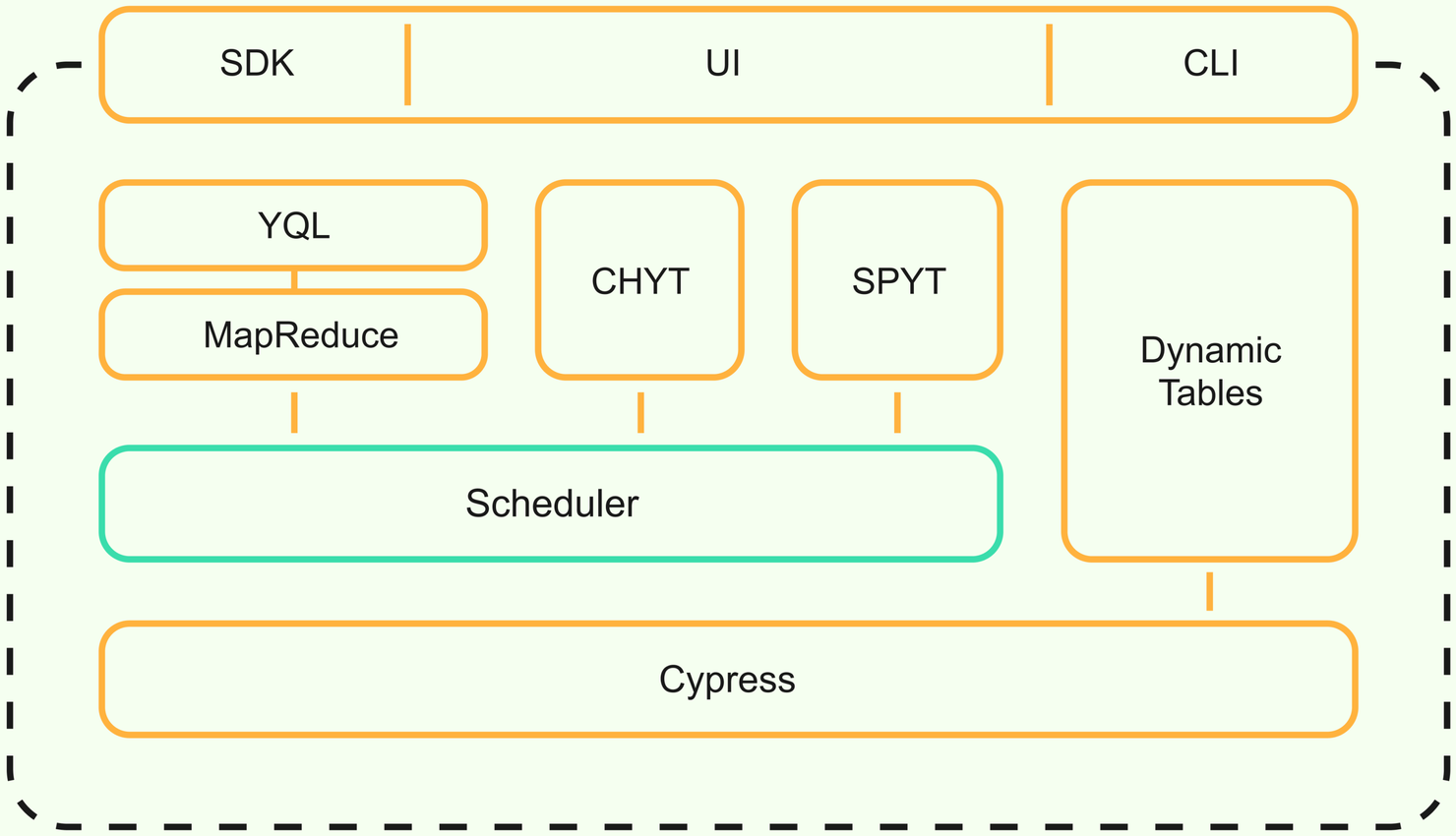
- YARN
- Mesos (Until 4.0.0)
- Kubernetes

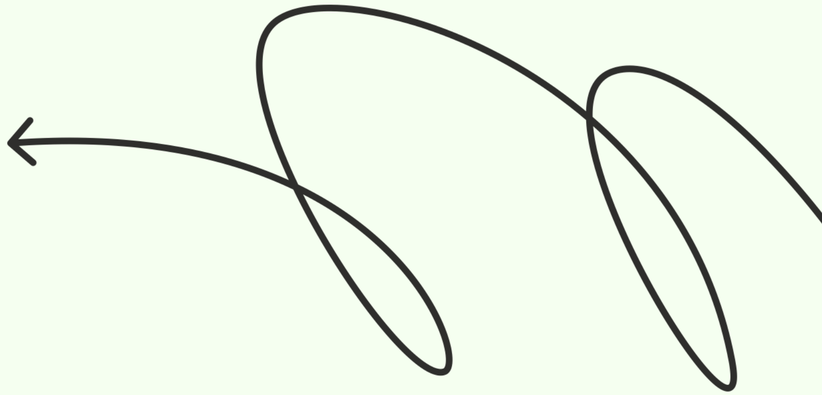
Реализации подключаемых кластер-менеджеров

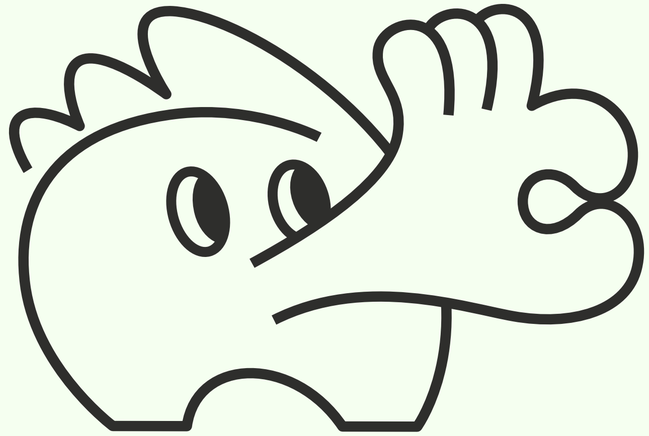


Spark + YTsaurus

Компоненты YTsaurus







YTsaurus Scheduler

- Отвечает за распределение ресурсов (cpu, memory, user_slots, network) между задачами

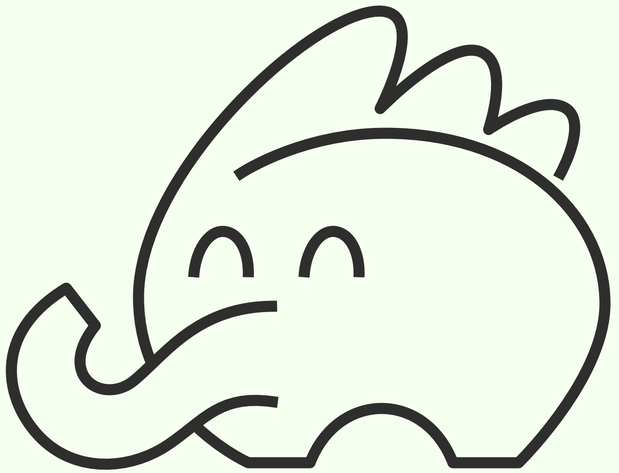
YTsaurus Scheduler

- Отвечает за распределение ресурсов (cpu, memory, user_slots, network) между задачами
- Определяет очередность, в которой эти задачи будут выполняться

YTsaurus Scheduler concepts

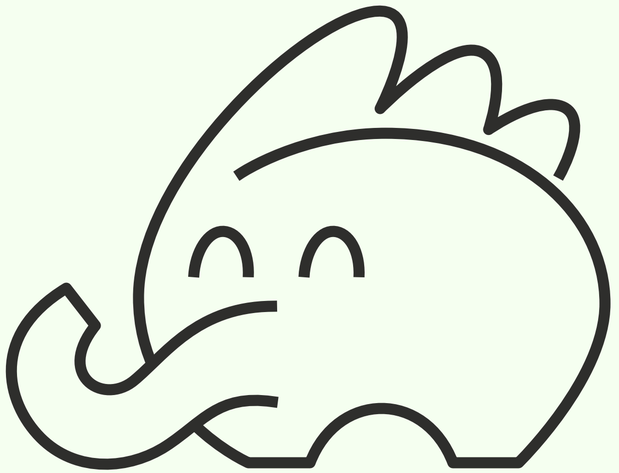
Типы операций в YTsaurus

- Map, Reduce, MapReduce — выполняют пользовательский код над входными данными



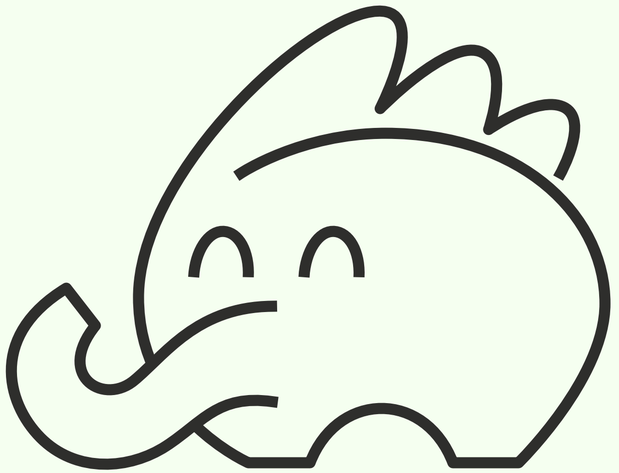
Типы операций в YTsaurus

- Map, Reduce, MapReduce — выполняют пользовательский код над входными данными
- Sort — сортирует указанную на входе таблицу



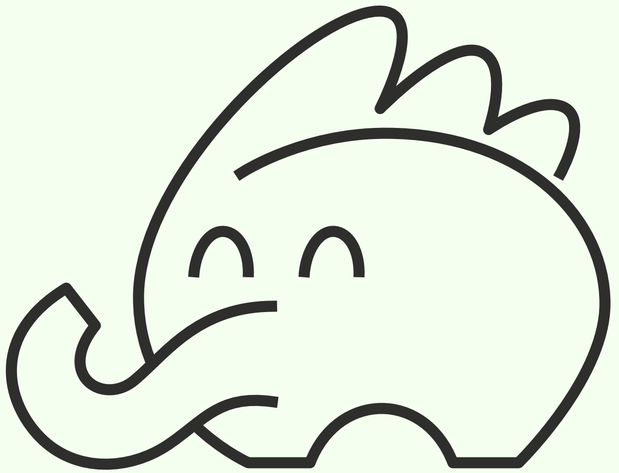
Типы операций в YTsaurus

- Map, Reduce, MapReduce — выполняют пользовательский код над входными данными
- Sort — сортирует указанную на входе таблицу
- Merge — выполняет слияние таблиц



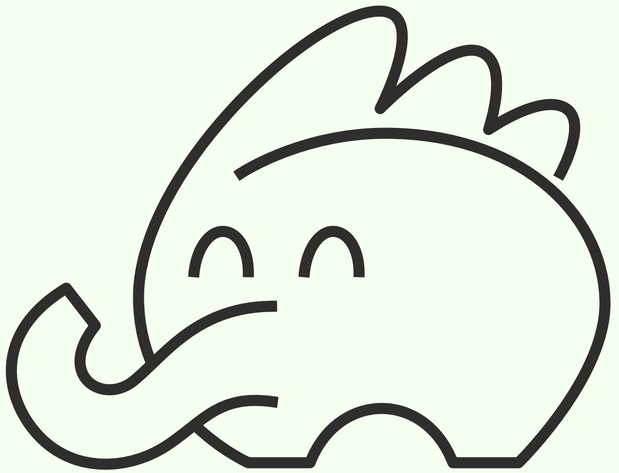
Типы операций в YTsaurus

- Map, Reduce, MapReduce — выполняют пользовательский код над входными данными
- Sort — сортирует указанную на входе таблицу
- Merge — выполняет слияние таблиц
- Erase — удаляет из таблицы указанный диапазон данных



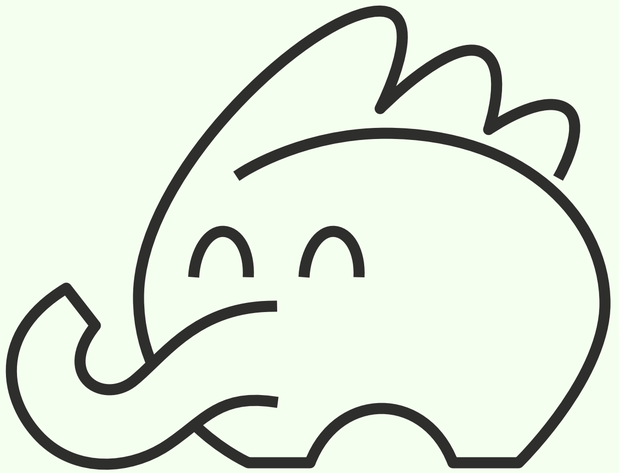
Типы операций в YTsaurus

- Map, Reduce, MapReduce — выполняют пользовательский код над входными данными
- Sort — сортирует указанную на входе таблицу
- Merge — выполняет слияние таблиц
- Erase — удаляет из таблицы указанный диапазон данных
- RemoteCopy — копирует данные между кластерами

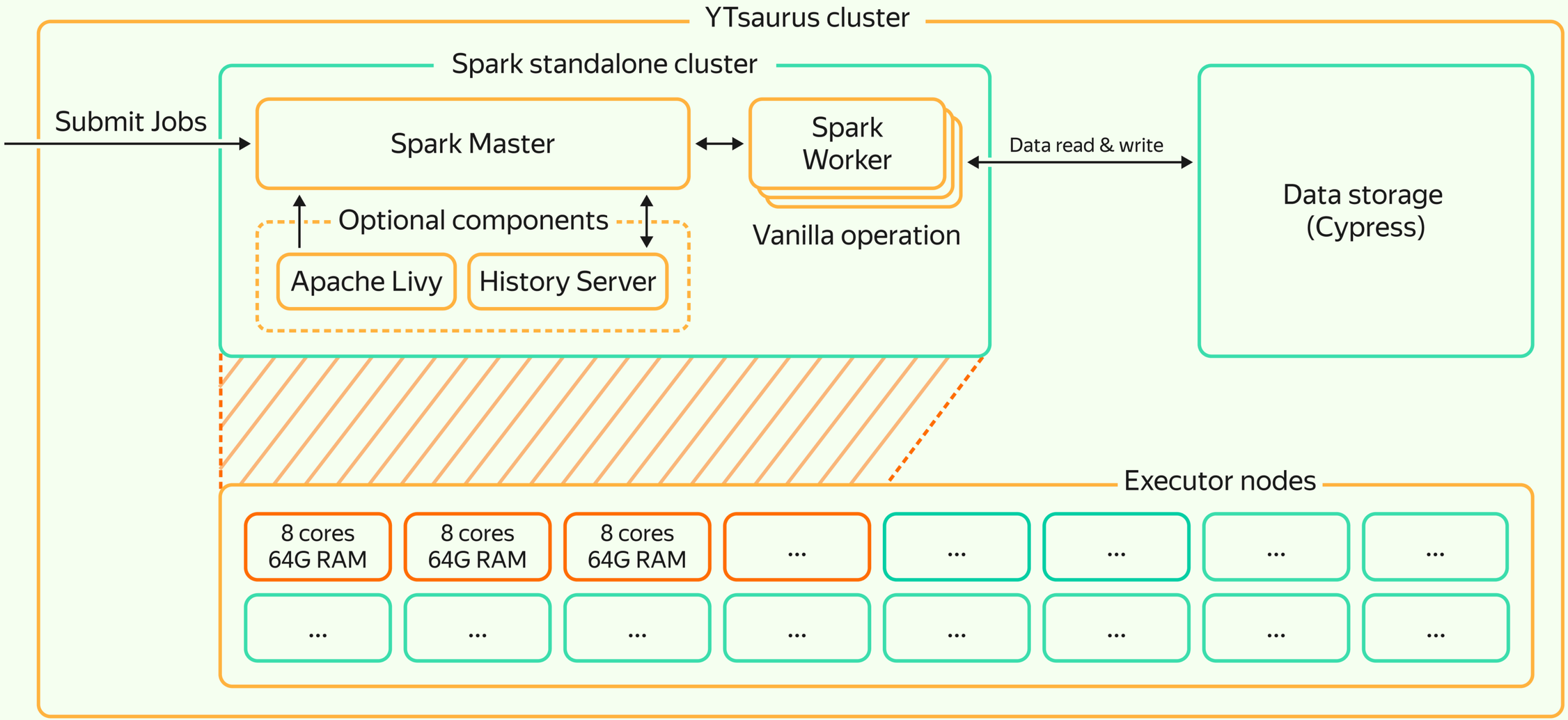


Типы операций в YTsaurus

- Map, Reduce, MapReduce — выполняют пользовательский код над входными данными
- Sort — сортирует указанную на входе таблицу
- Merge — выполняет слияние таблиц
- Erase — удаляет из таблицы указанный диапазон данных
- RemoteCopy — копирует данные между кластерами
- Vanilla — запускает набор пользовательских скриптов в нужном количестве на узлах кластера и поддерживает их жизнедеятельность



Как запускали Spark-задачи в YTsaurus раньше



Недостатки такого подхода

- Ресурсы постоянно зарезервированы, но не всегда выполняют полезную работу

Недостатки такого подхода

- Ресурсы постоянно зарезервированы, но не всегда выполняют полезную работу
- Двойное управление ресурсами: на уровне YTsaurus и на уровне внутреннего Standalone-кластера

Недостатки такого подхода

- Ресурсы постоянно зарезервированы, но не всегда выполняют полезную работу
- Двойное управление ресурсами: на уровне YTsaurus и на уровне внутреннего Standalone-кластера
- Схема эффективна только при 100% загрузке внутреннего кластера, на практике загрузка не выше 10%

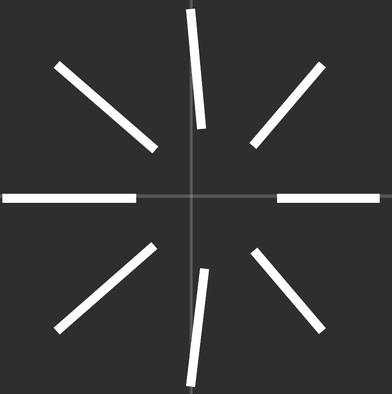
Недостатки такого подхода

- Ресурсы постоянно зарезервированы, но не всегда выполняют полезную работу
- Двойное управление ресурсами: на уровне YTsaurus и на уровне внутреннего Standalone-кластера
- Схема эффективна только при 100% загрузке внутреннего кластера, на практике загрузка не выше 10%
- Для разовых расчётов необходимо сначала поднять кластер, и потом запустить задачу

Недостатки такого подхода

- Ресурсы постоянно зарезервированы, но не всегда выполняют полезную работу
- Двойное управление ресурсами: на уровне YTsaurus и на уровне внутреннего Standalone-кластера
- Схема эффективна только при 100% загрузке внутреннего кластера, на практике загрузка не выше 10%
- Для разовых расчётов необходимо сначала поднять кластер, и потом запустить задачу
- Сложно мигрировать с других кластеров, например, с хадупа так, как для запуска задач используются обёртки над стандартными утилитами спарка

Почему бы не сделать свою реализацию cluster manager по аналогии с уже существующими?



Кластер менеджеры в Spark

Встроенные:

- Local
- Standalone

Кластер менеджеры в Spark

Встроенные:

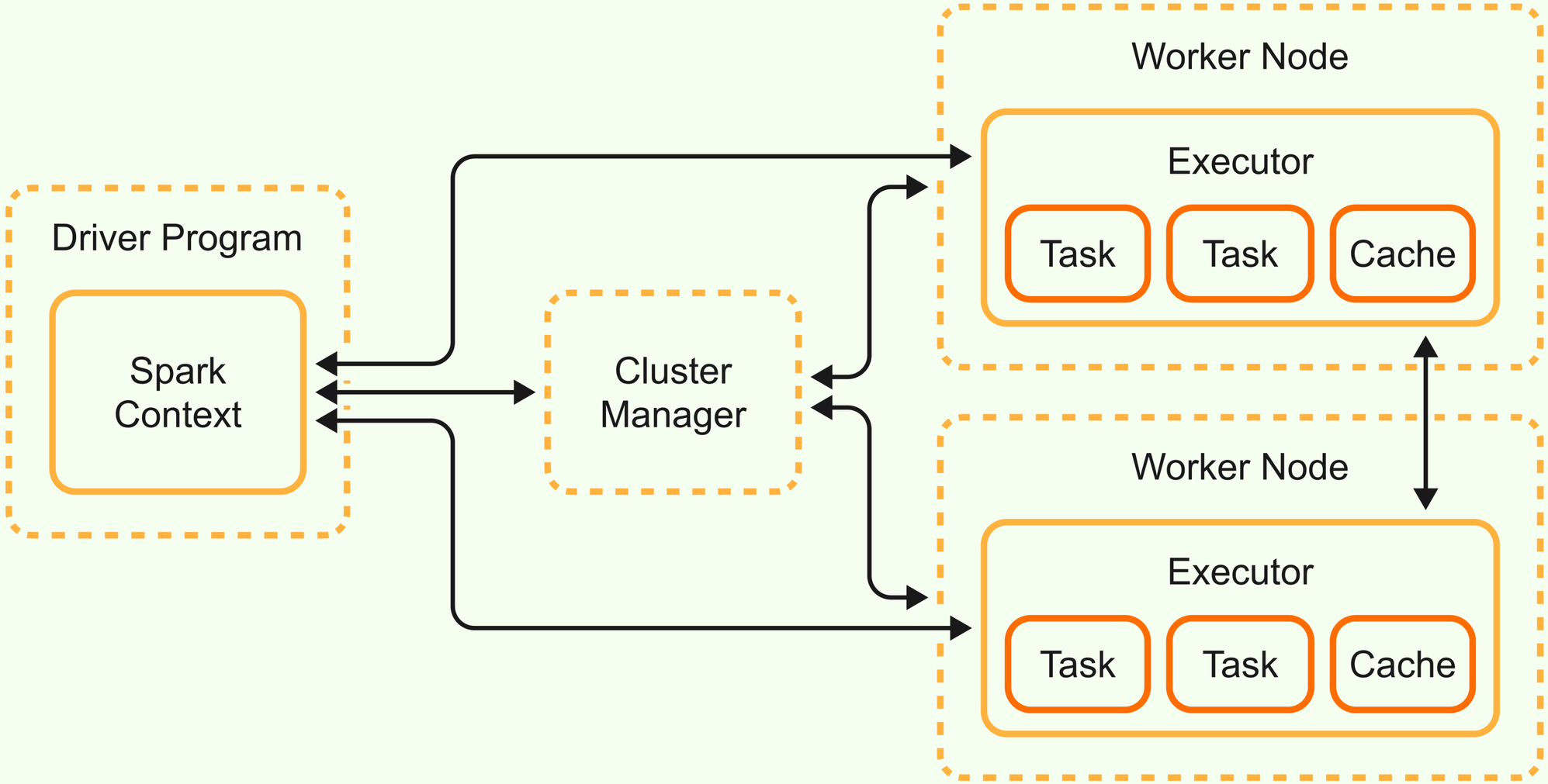
- Local
- Standalone

Подключаемые:

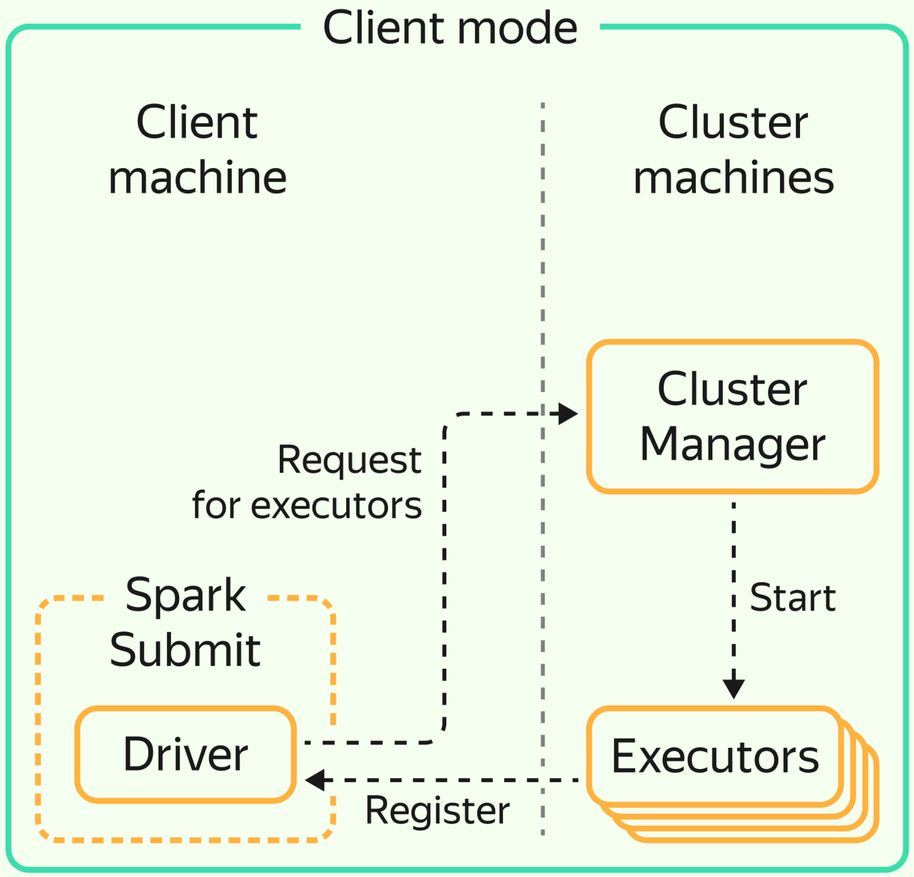
- YARN
- Mesos (Until 4.0.0)
- Kubernetes
- + YTsaurus !!!

Анатомия Spark-приложения

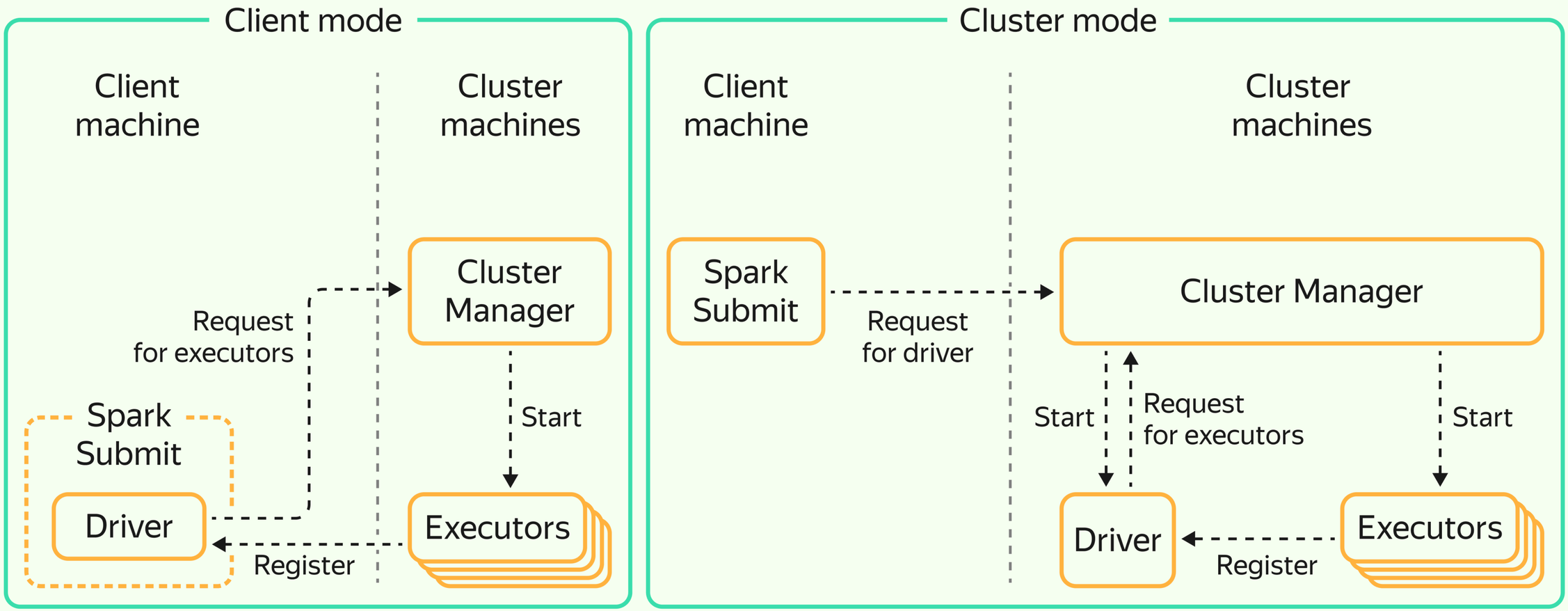
Как работает Spark-приложение?



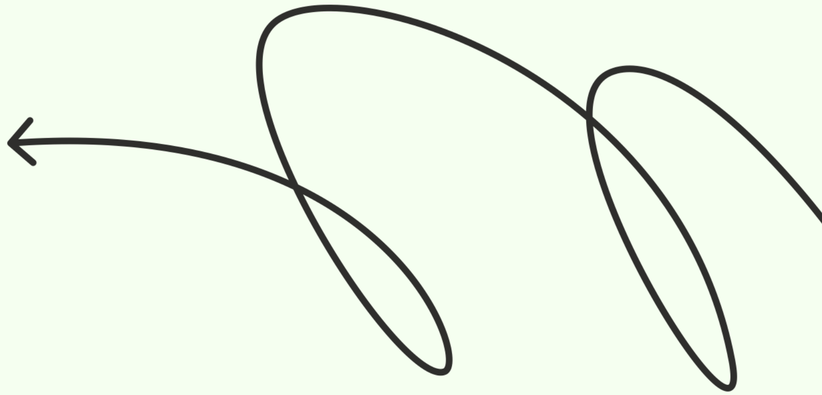
Взаимодействие Spark-процессов



Взаимодействие Spark-процессов



Три вида процессов в Spark

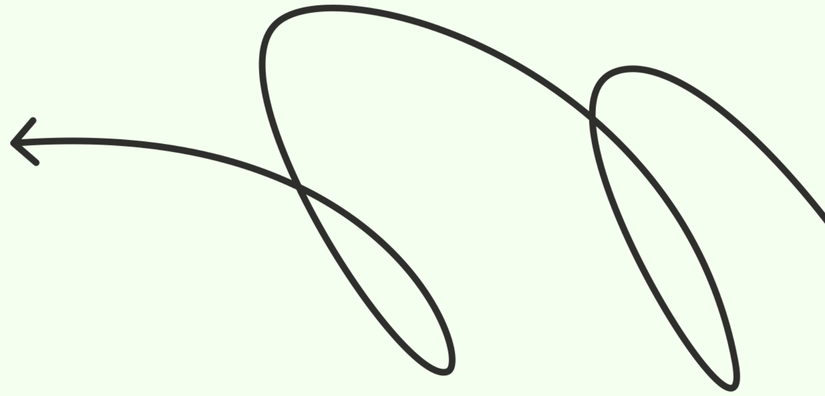


1

Driver

Управление запуском
и оркестрация
распределенного
процесса

Три вида процессов в Spark



1

Driver

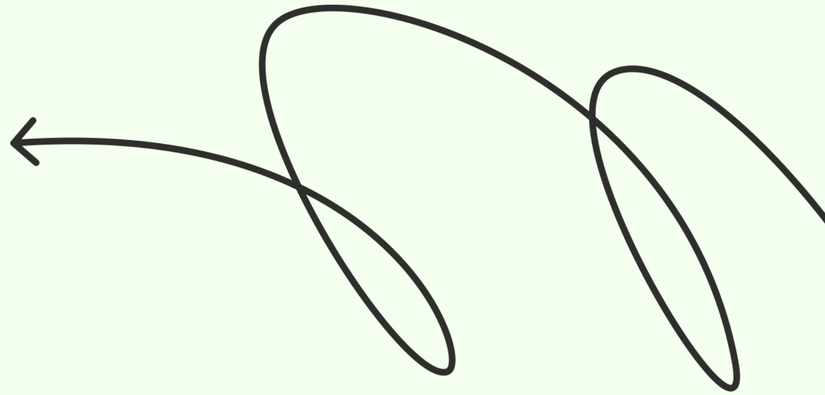
Управление запуском и оркестрация распределенного процесса

2

Executor

Непосредственное выполнение подзадач распределенного вычисления

Три вида процессов в Spark



1

Driver

Управление запуском и оркестрация распределенного процесса

2

Executor

Непосредственное выполнение подзадач распределенного вычисления

3

SparkSubmit

Запуск spark-приложения

Постановка задачи

Что нужно для подключения

Цель: Научиться запускать все три компонента приложения используя новый resource manager

Что нужно для подключения

Цель: Научиться запускать все три компонента приложения используя новый resource manager

Для этого:

- Добавить модуль с реализацией в resource-managers

Что нужно для подключения

Цель: Научиться запускать все три компонента приложения используя новый resource manager

Для этого:

- Добавить модуль с реализацией в resource-managers
- Сделать реализации основных компонентов SPI

Что нужно для подключения

Цель: Научиться запускать все три компонента приложения используя новый resource manager

Для этого:

- Добавить модуль с реализацией в resource-managers
- Сделать реализации основных компонентов SPI
- Сделать модификации класса SparkSubmit в модуле core

Что нужно для подключения

Цель: Научиться запускать все три компонента приложения используя новый resource manager

Для этого:

- Добавить модуль с реализацией в resource-managers
- Сделать реализации основных компонентов SPI
- Сделать модификации класса SparkSubmit в модуле core
- …

Что нужно для подключения

Цель: Научиться запускать все три компонента приложения используя новый resource manager

Для этого:

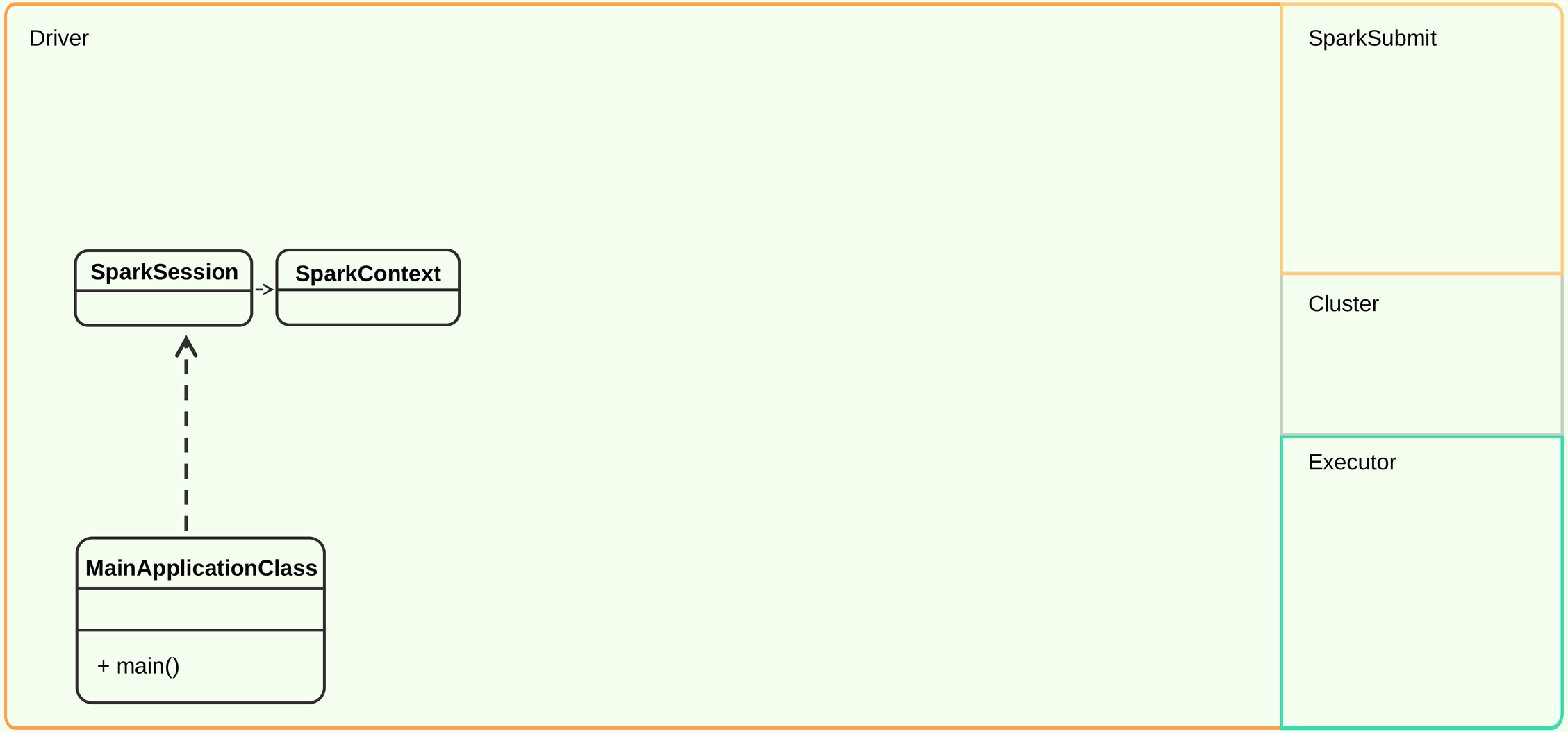
- Добавить модуль с реализацией в resource-managers
- Сделать реализации основных компонентов SPI
- Сделать модификации класса SparkSubmit в модуле core
- …
- PROFIT!

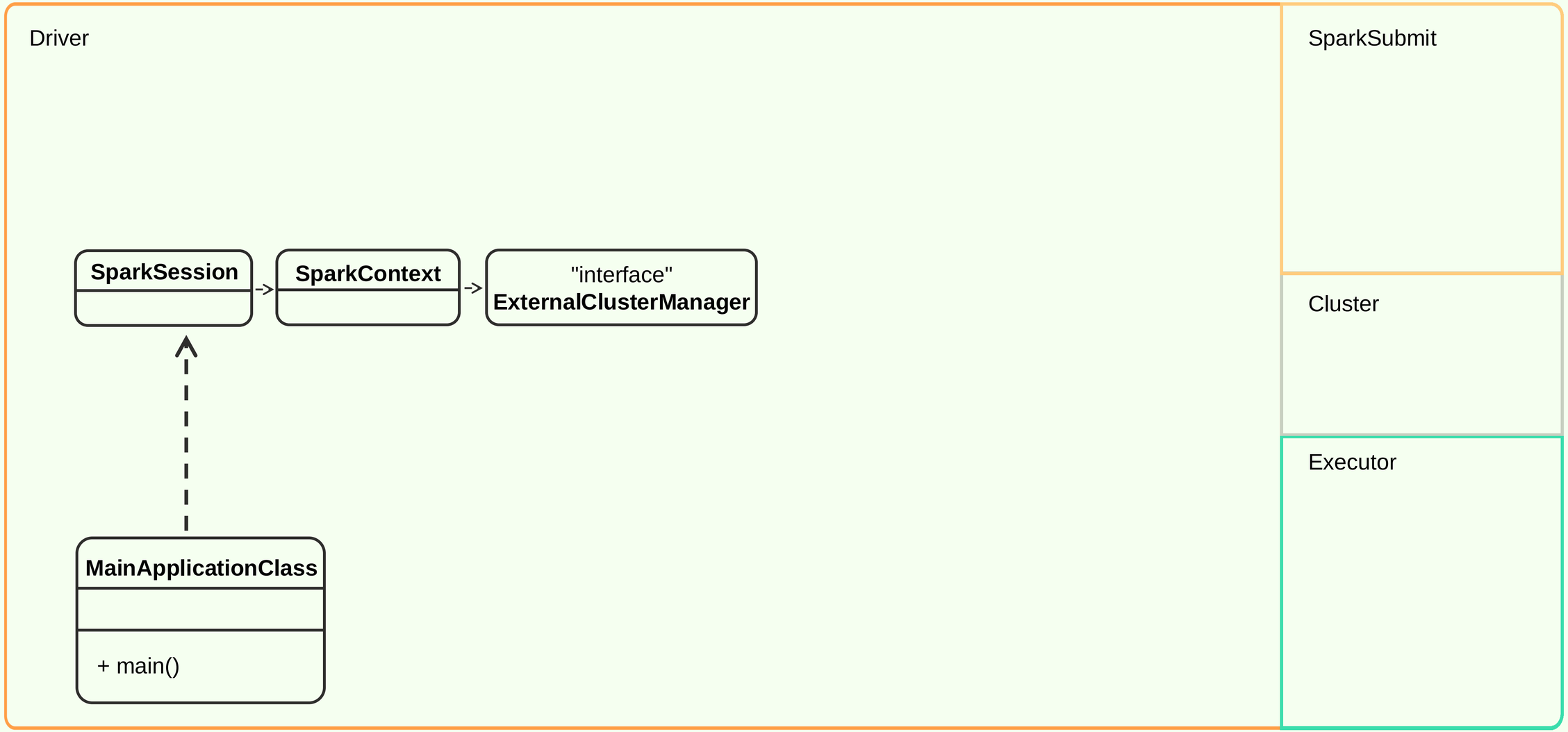
SPI для Spark resource manager

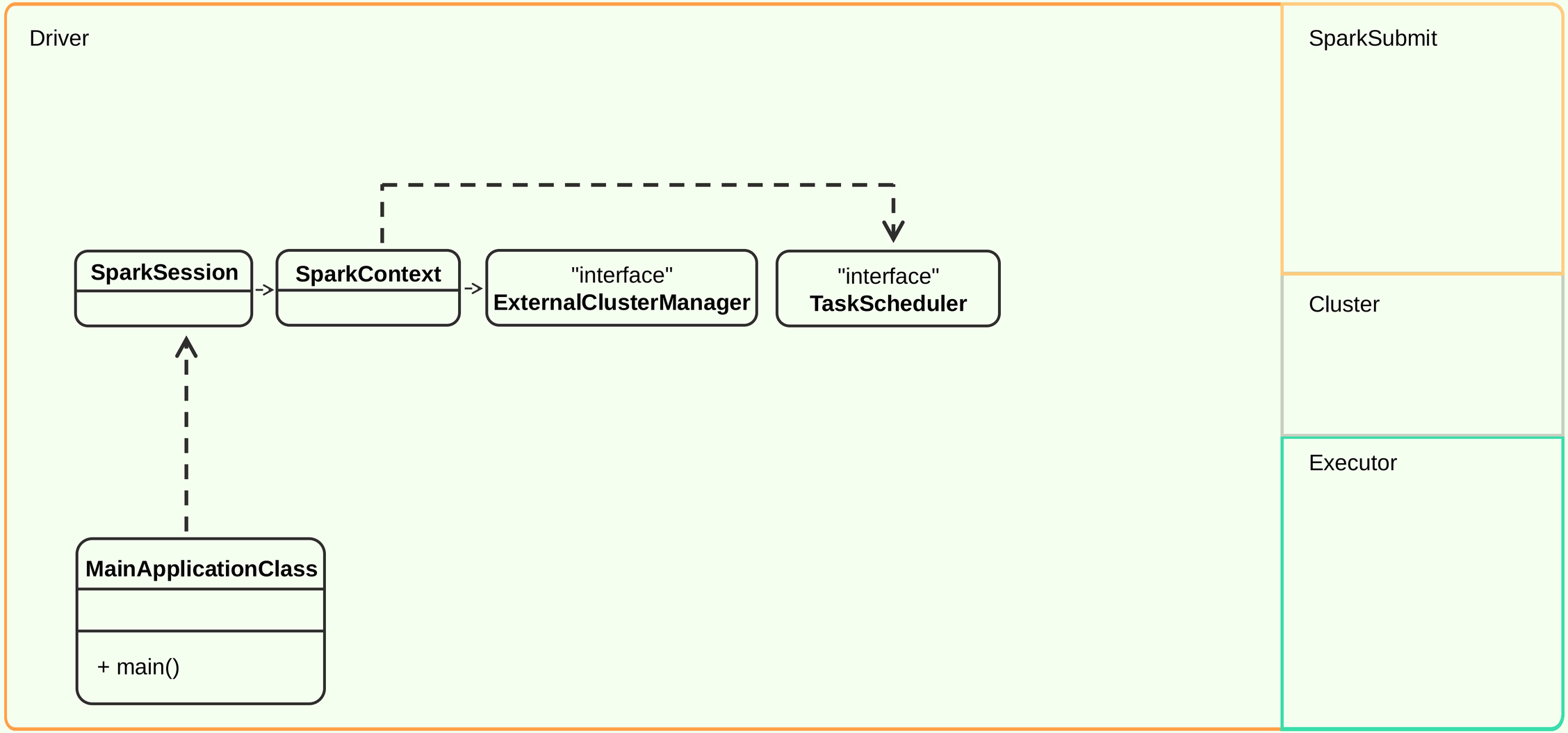

Driver
SparkSubmit
Cluster
Executor

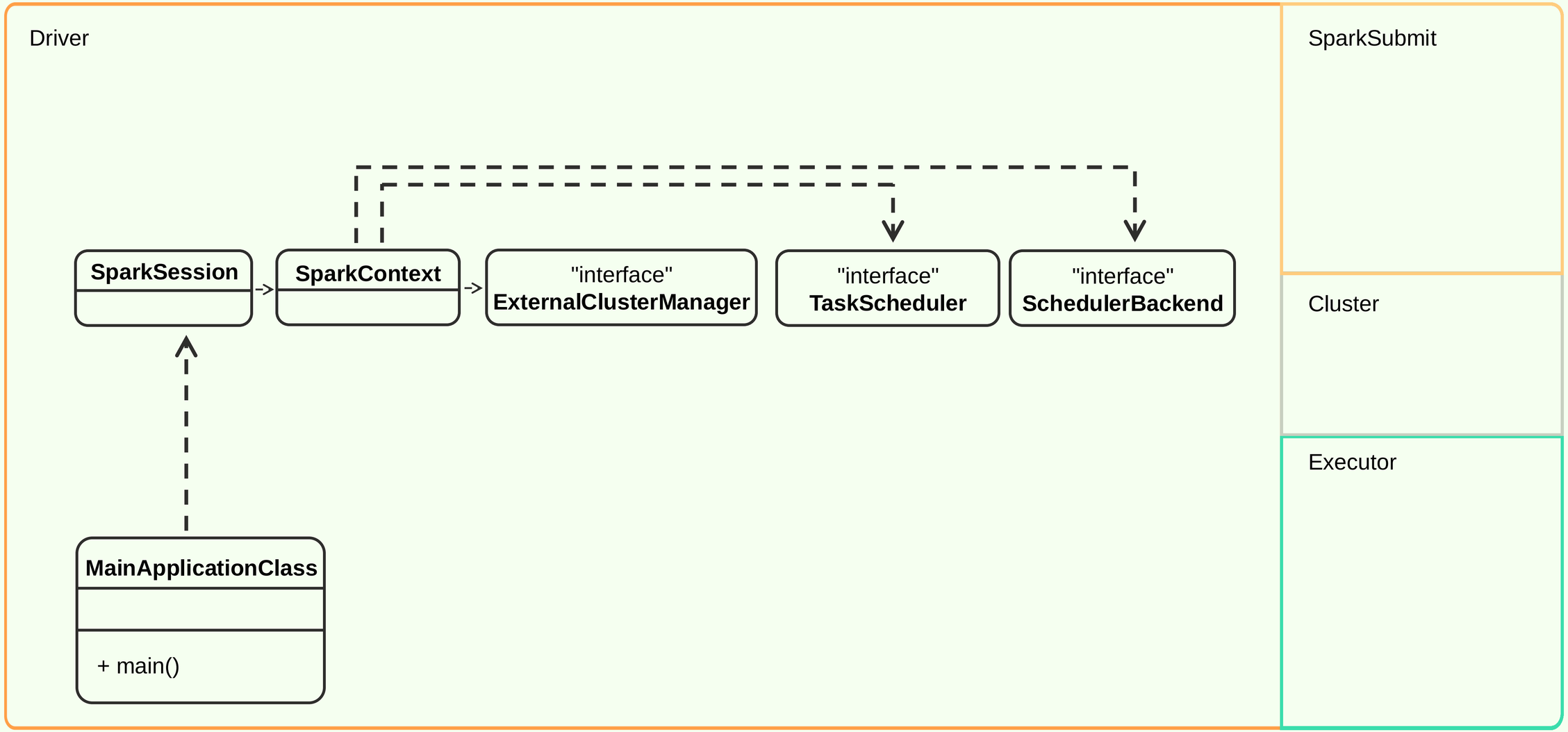

Driver
SparkSubmit
Cluster
Executor
MainApplicationClass
+ main()

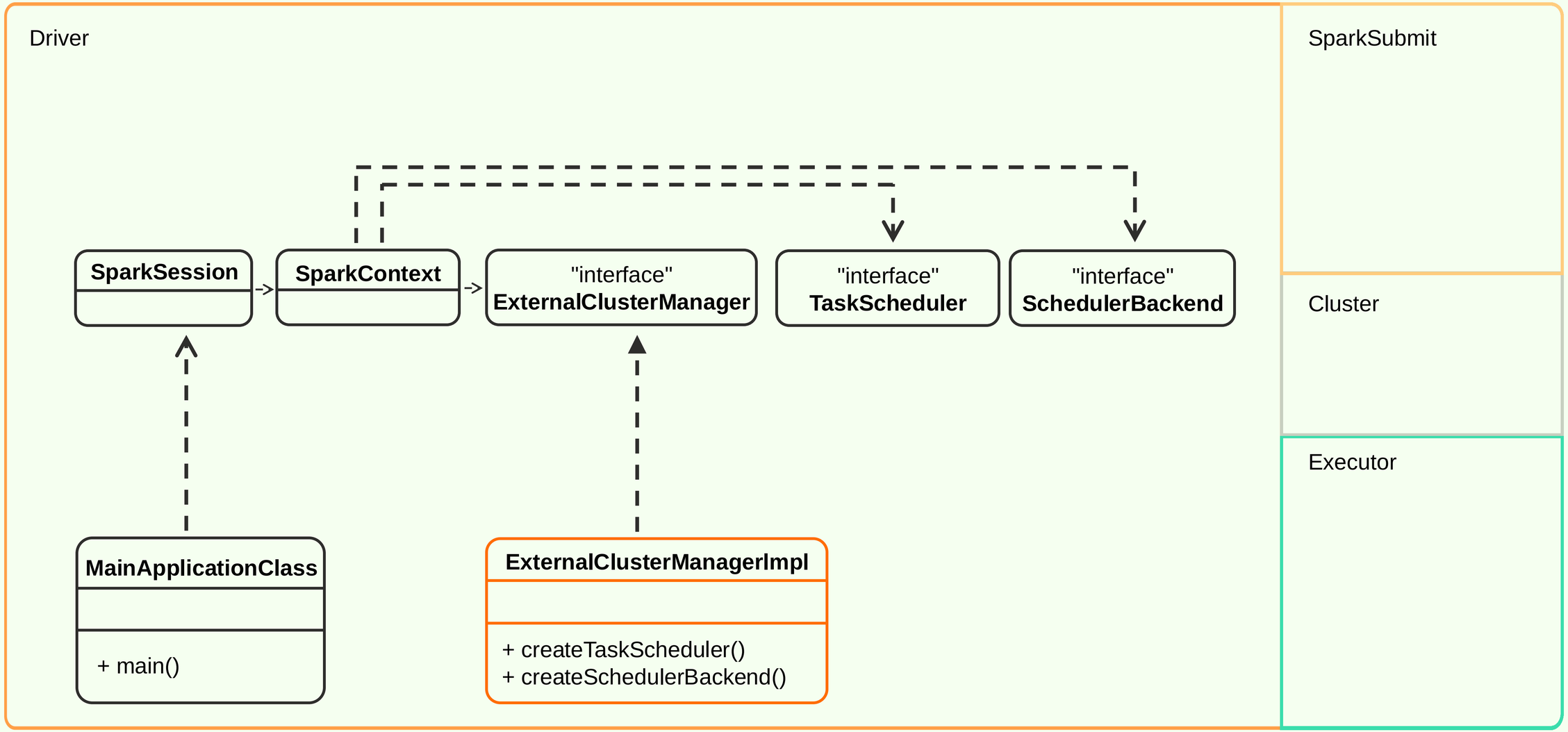

Driver
SparkSession
MainApplicationClass
+ main()
SparkSubmit
Cluster
Executor

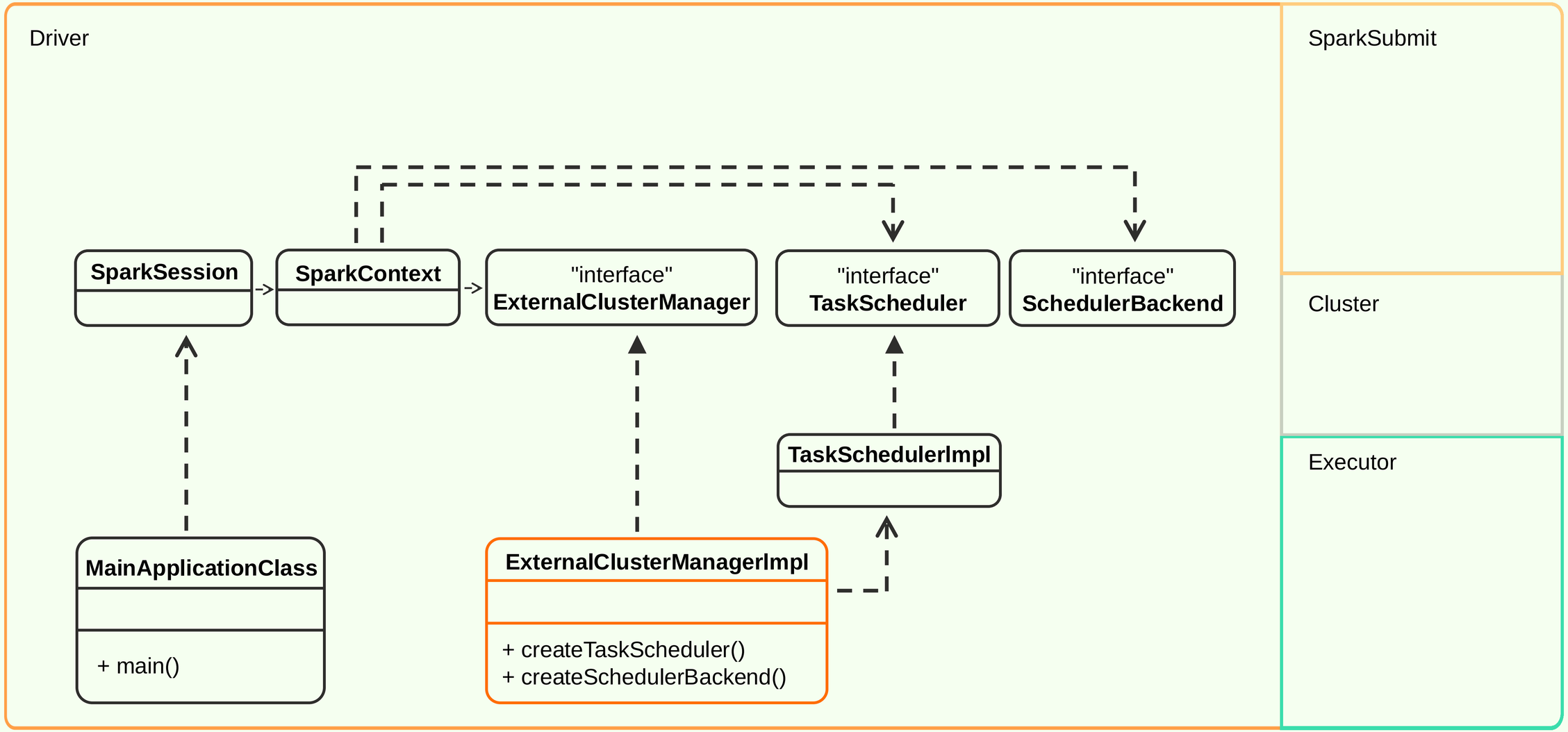

Driver
SparkSession
->
SparkContext
MainApplicationClass
+ main()
SparkSubmit
Cluster
Executor

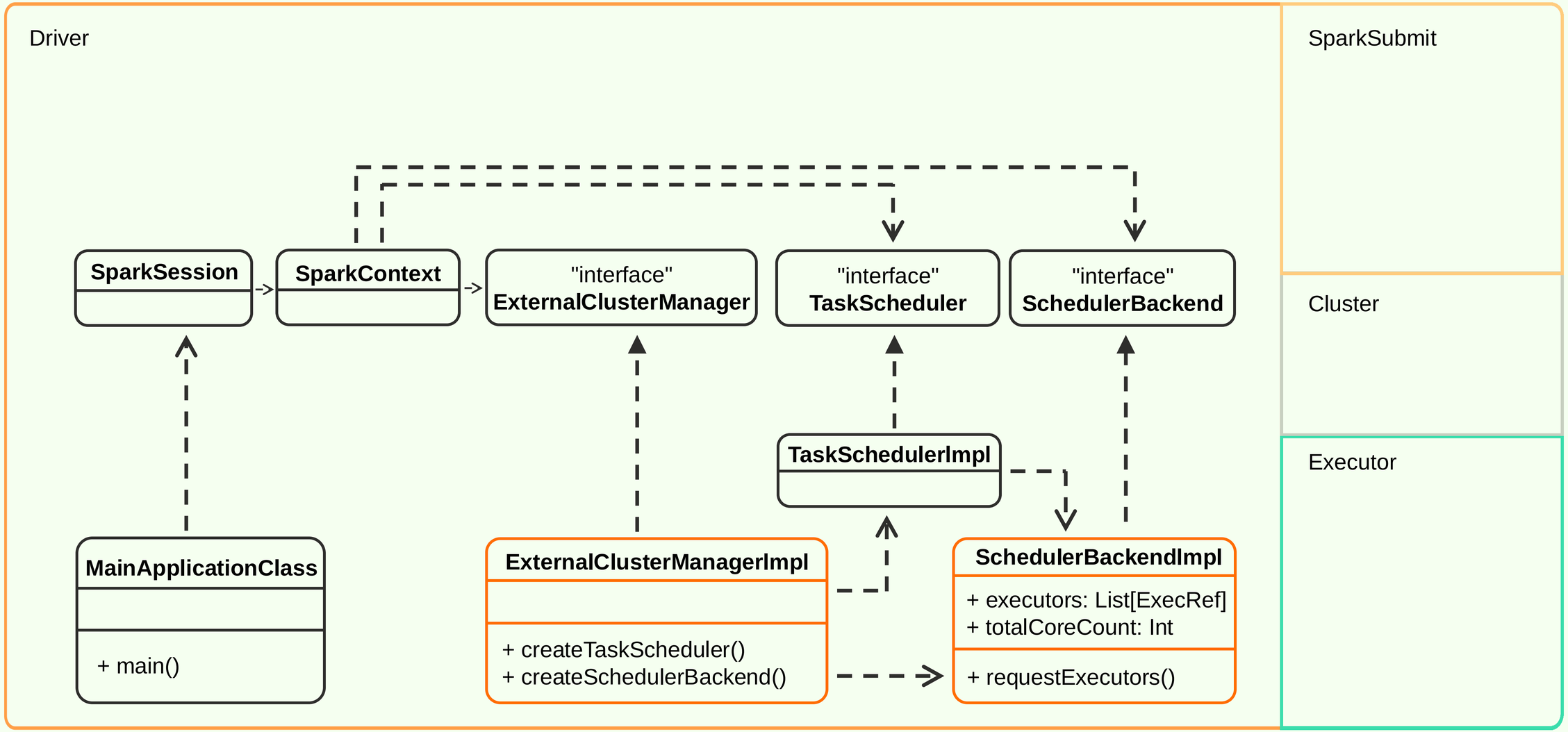

Driver
SparkSubmit
SparkSession
->
SparkContext
->
"interface"
ExternalClusterManager
Cluster
Executor
MainApplicationClass
+ main()

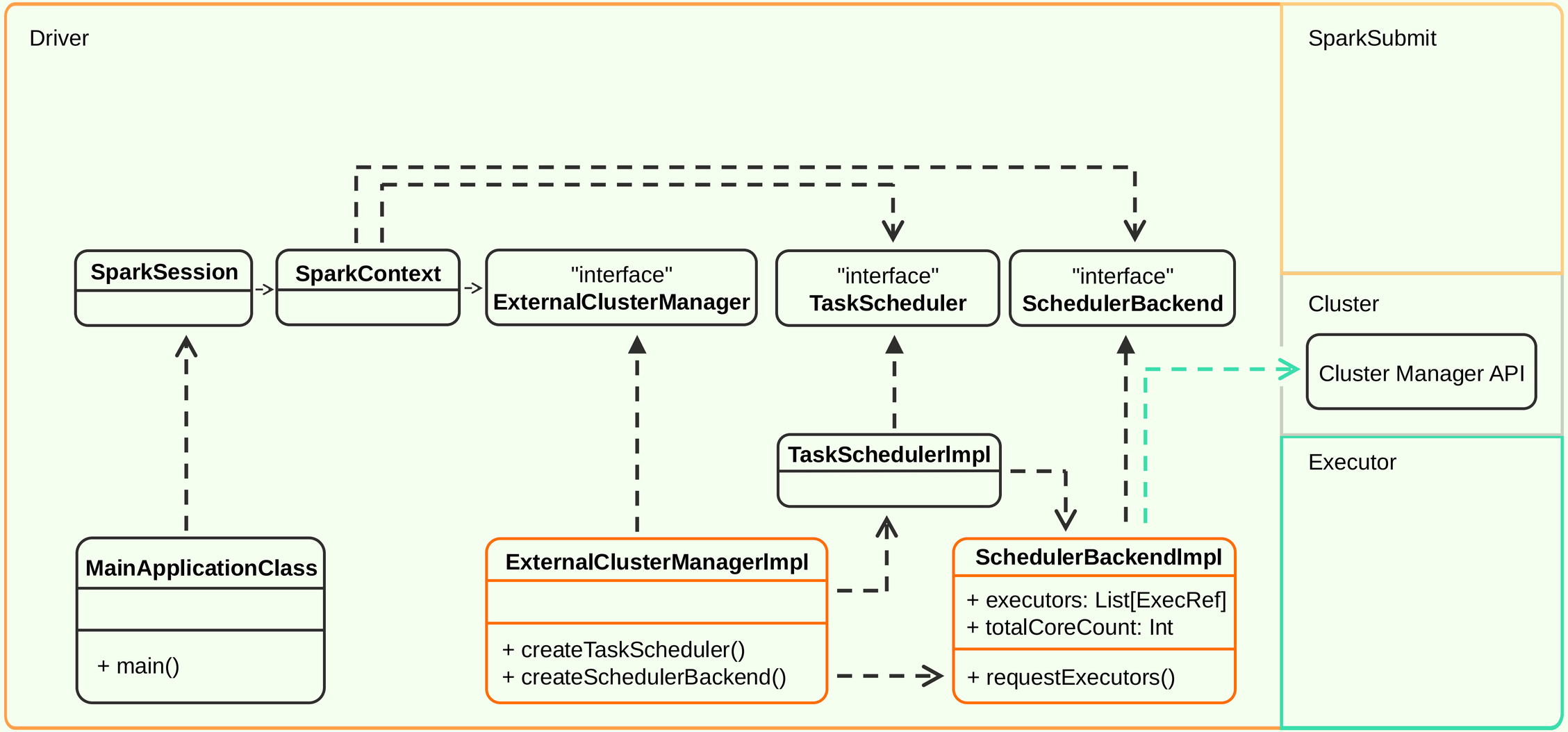

Driver
SparkSession
SparkContext
"interface"
ExternalClusterManager
"interface"
TaskScheduler
MainApplicationClass
+ main()
SparkSubmit
Cluster
Executor

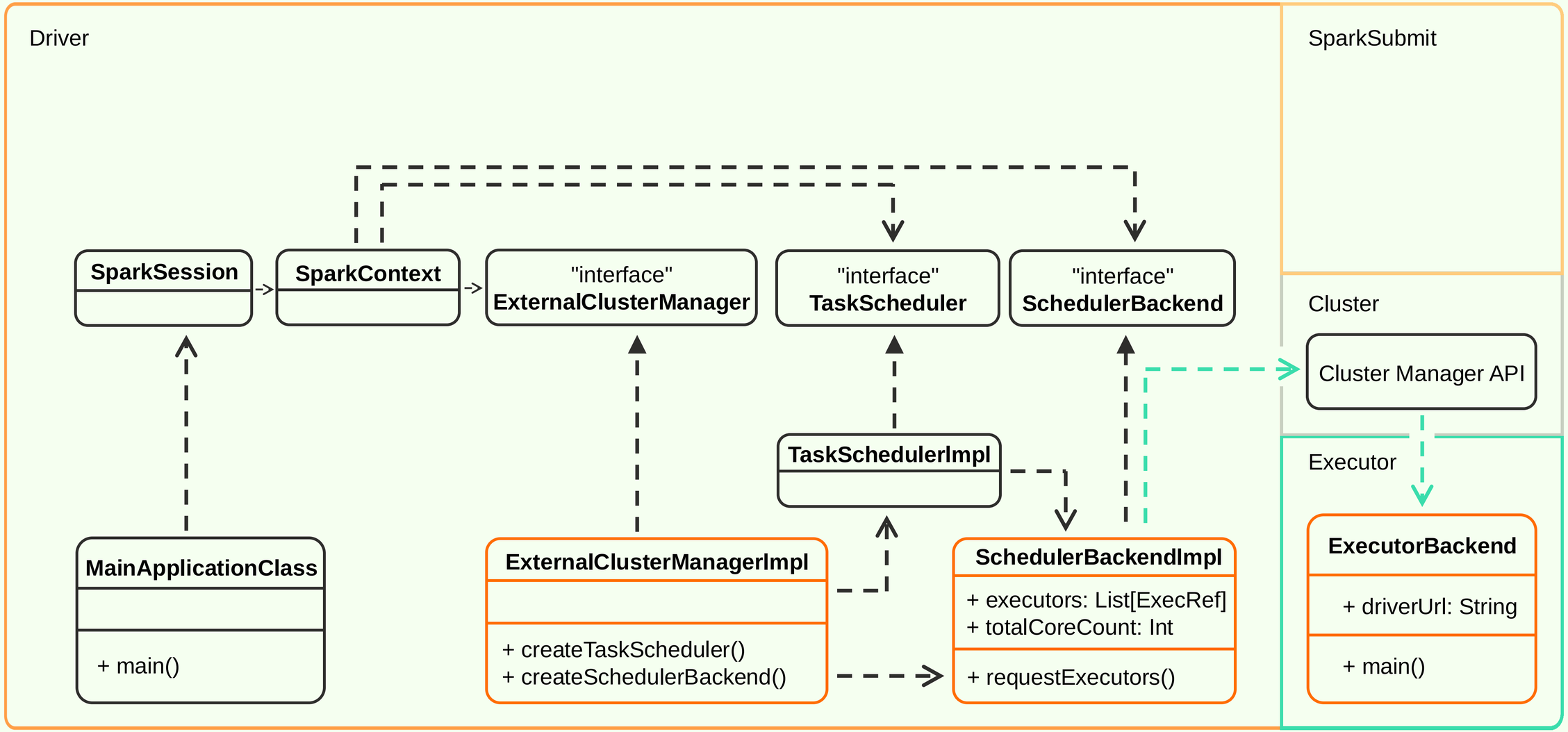

Driver
SparkSubmit
Cluster
Executor
SparkSession
SparkContext
"interface"
ExternalClusterManager
"interface"
TaskScheduler
"interface"
SchedulerBackend
MainApplicationClass
+ main()

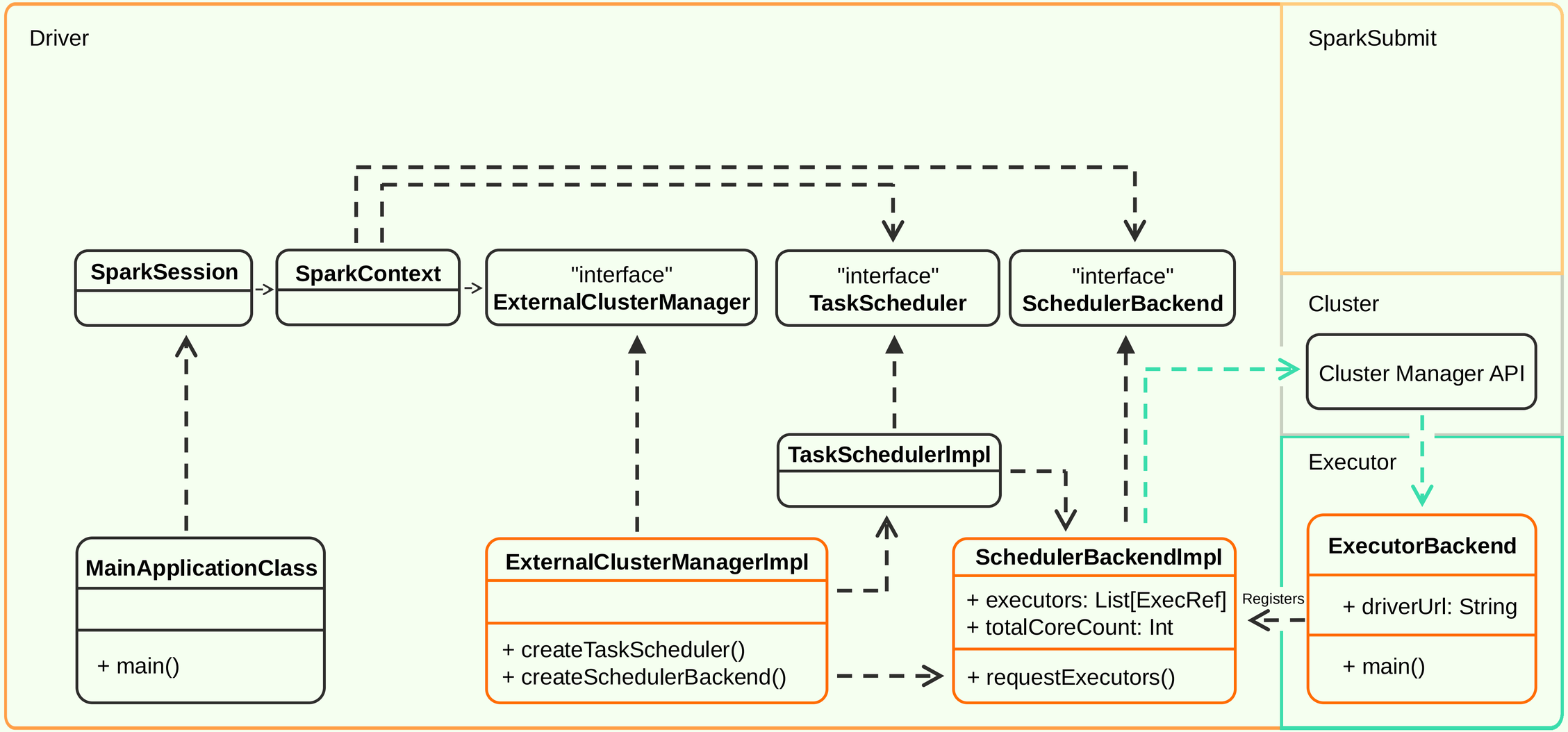

Driver
SparkSubmit
Cluster
Executor
SparkSession
SparkContext
"interface"
ExternalClusterManager
"interface"
TaskScheduler
"interface"
SchedulerBackend
MainApplicationClass
+ main()
ExternalClusterManagerImpl
+ createTaskScheduler()
+ createSchedulerBackend()

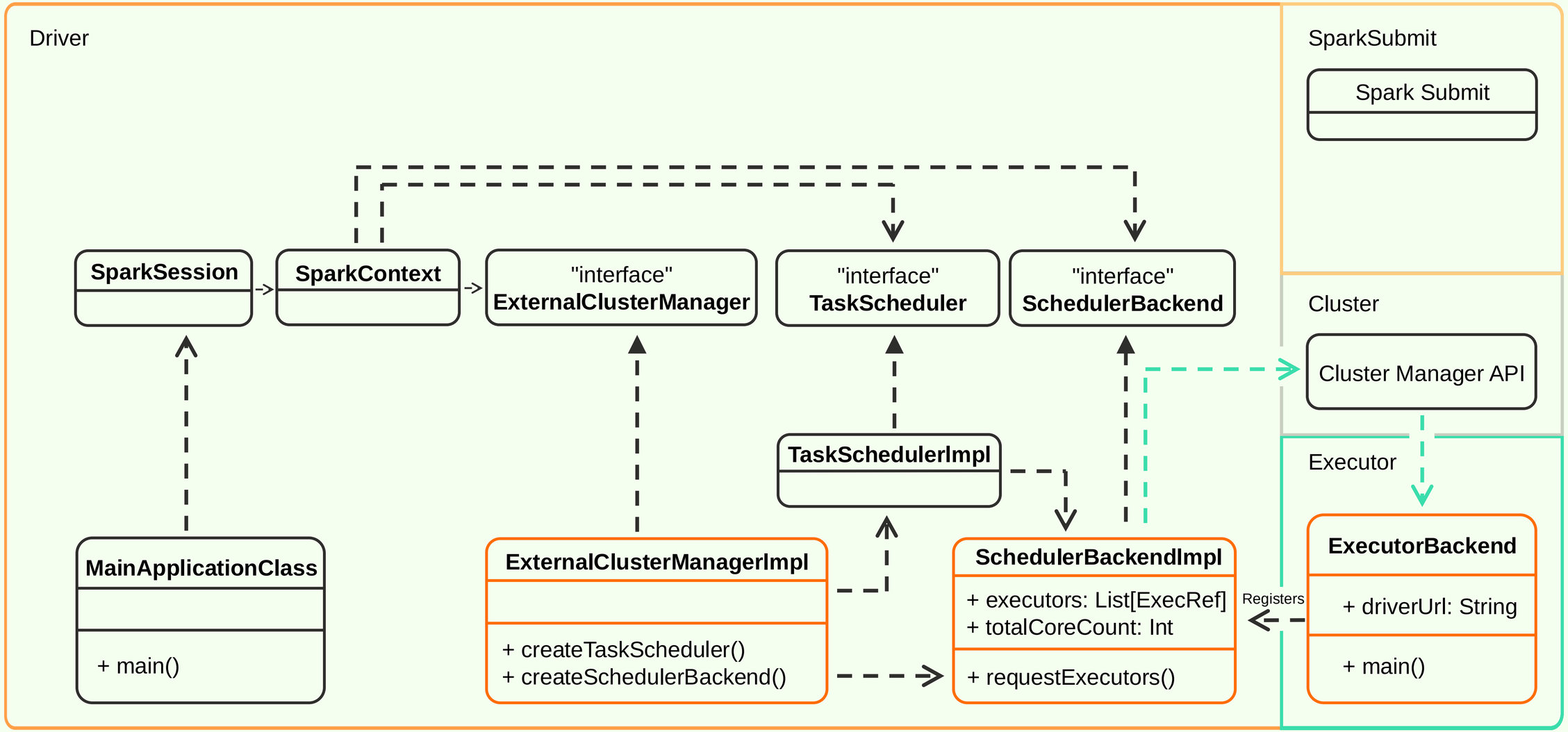

Driver
SparkSubmit
Cluster
Executor
SparkSession
SparkContext
"interface"
ExternalClusterManager
"interface"
TaskScheduler
"interface"
SchedulerBackend
TaskSchedulerImpl
MainApplicationClass
+ main()
ExternalClusterManagerImpl
+ createTaskScheduler()
+ createSchedulerBackend()

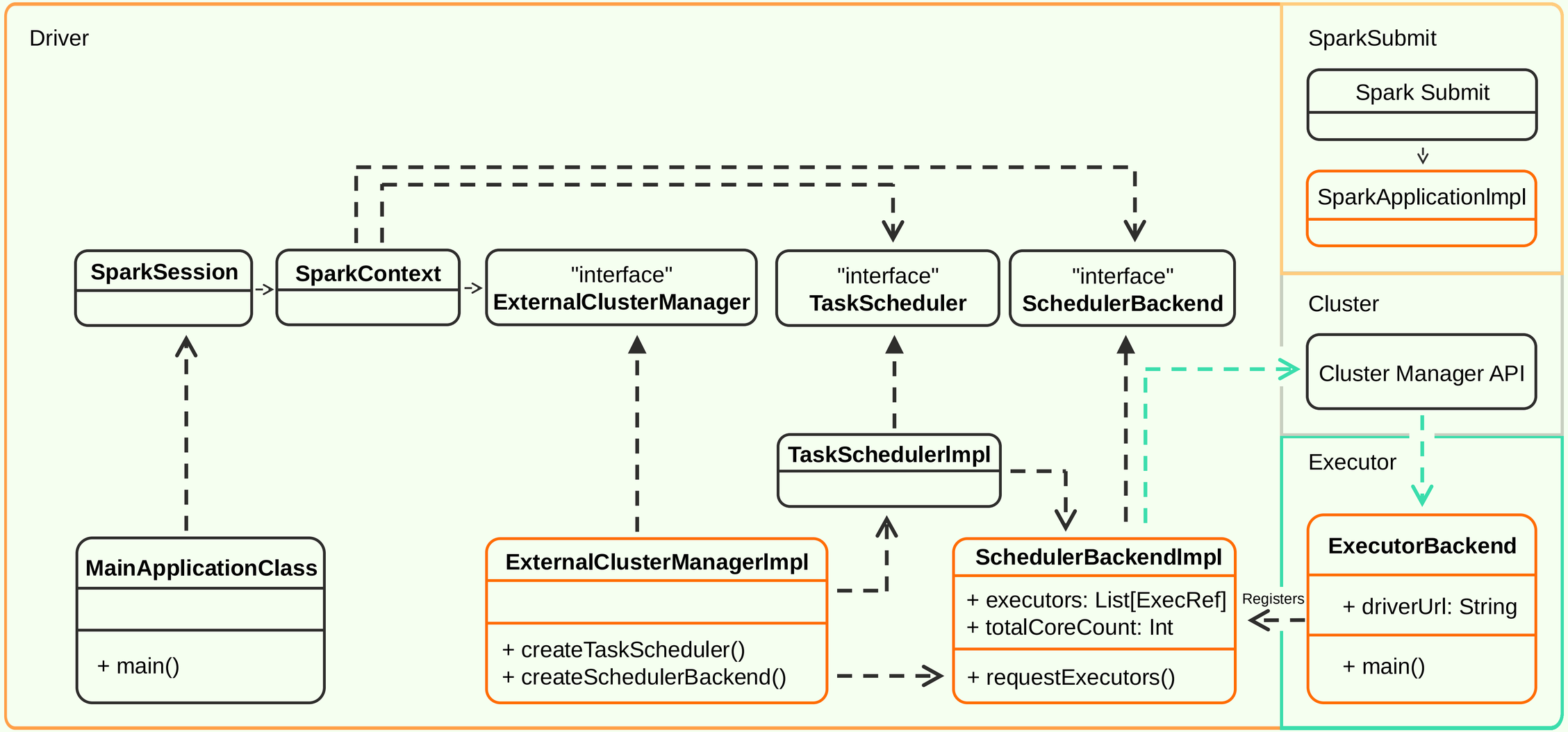

Driver
SparkSubmit
Cluster
Executor
SparkSession
SparkContext
"interface"
ExternalClusterManager
"interface"
TaskScheduler
"interface"
SchedulerBackend
TaskSchedulerImpl
MainApplicationClass
+ main()
ExternalClusterManagerImpl
+ createTaskScheduler()
+ createSchedulerBackend()
SchedulerBackendImpl
+ executors: List[ExecRef]
+ totalCoreCount: Int
+ requestExecutors()

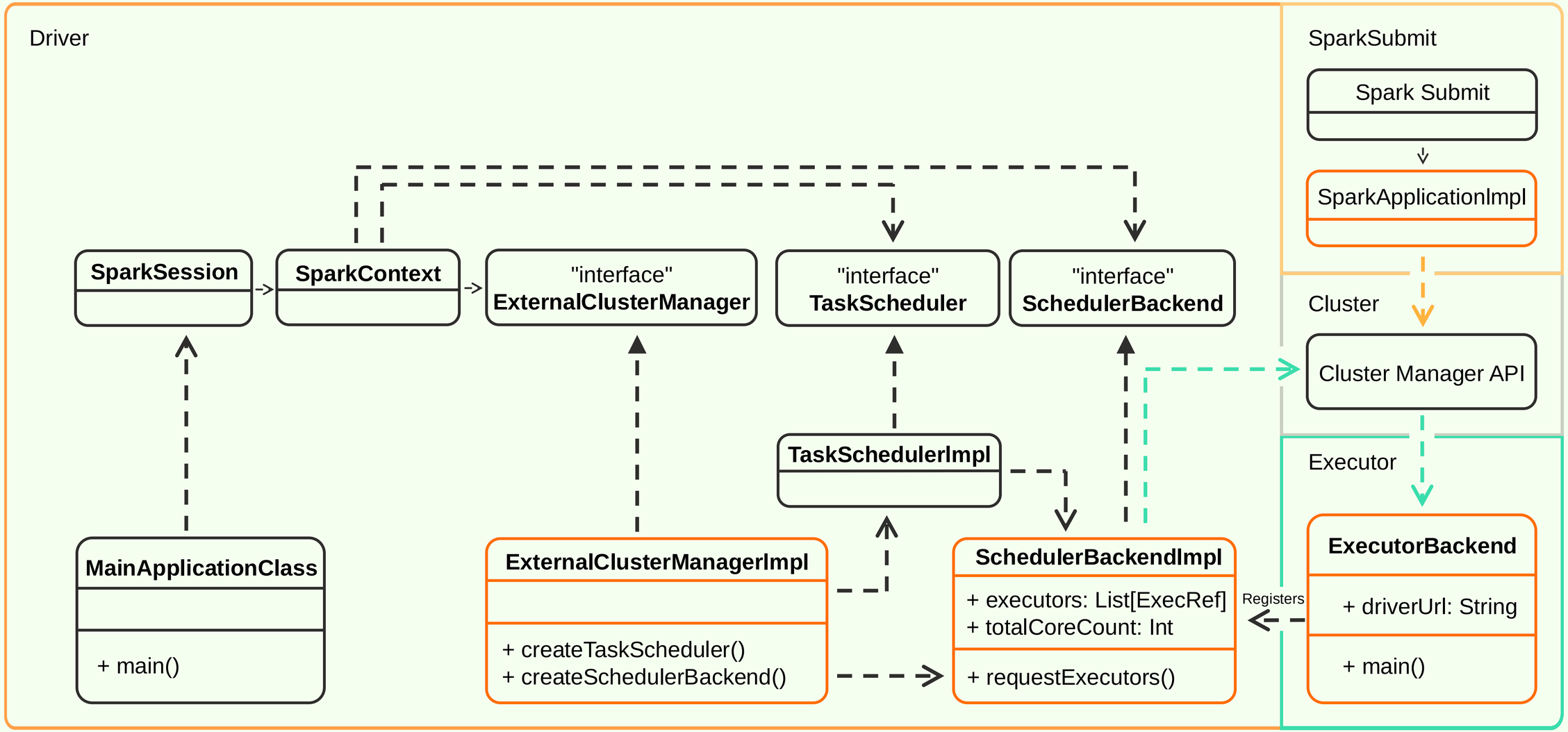

Driver
SparkSubmit
SparkSession
SparkContext
"interface"
ExternalClusterManager
"interface"
TaskScheduler
"interface"
SchedulerBackend
Cluster
Cluster Manager API
TaskSchedulerImpl
Executor
MainApplicationClass
+ main()
ExternalClusterManagerImpl
+ createTaskScheduler()
+ createSchedulerBackend()
SchedulerBackendImpl
+ executors: List[ExecRef]
+ totalCoreCount: Int
+ requestExecutors()

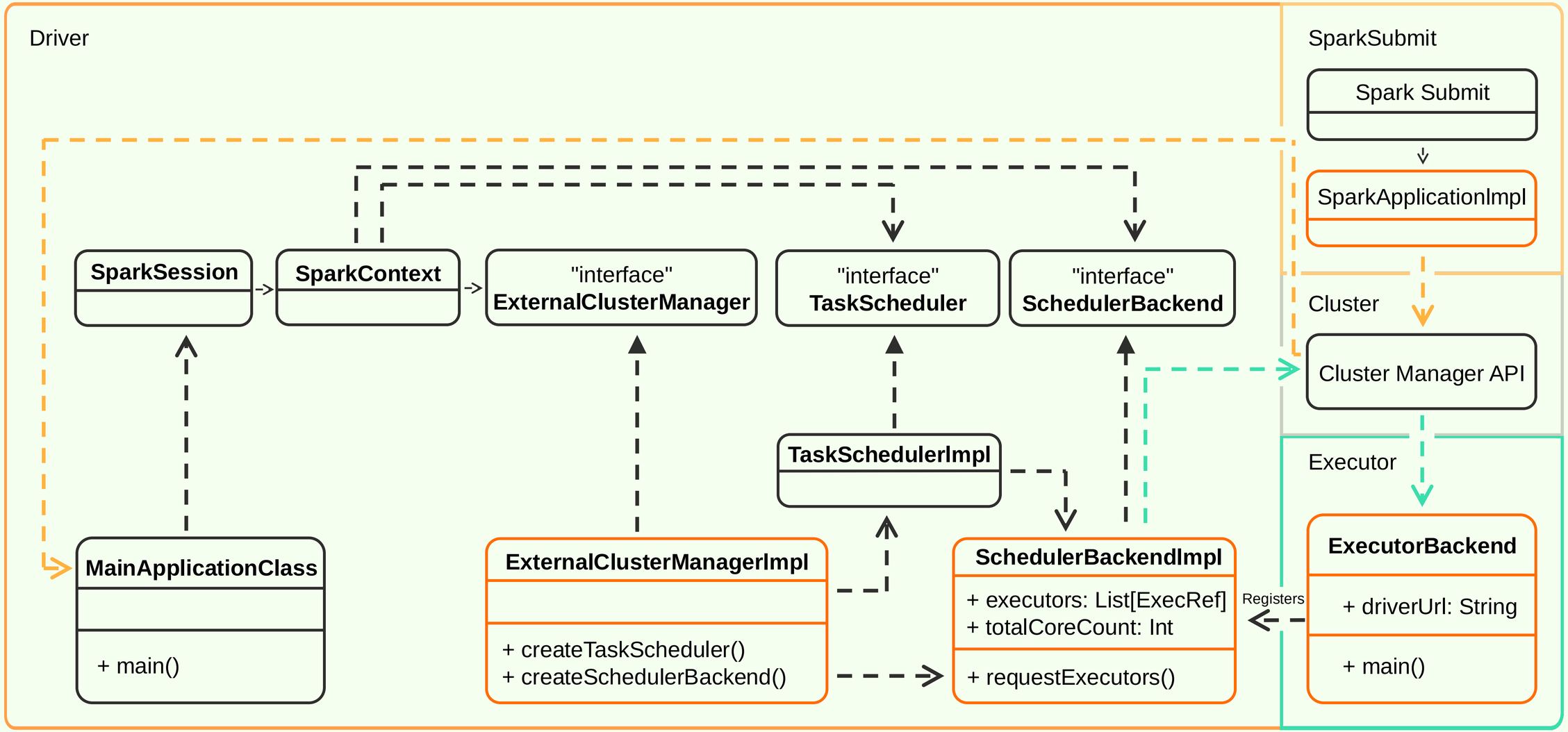

Driver
SparkSubmit
SparkSession
SparkContext
"interface"
ExternalClusterManager
"interface"
TaskScheduler
"interface"
SchedulerBackend
Cluster
Cluster Manager API
TaskSchedulerImpl
Executor
MainApplicationClass
+ main()
ExternalClusterManagerImpl
+ createTaskScheduler()
+ createSchedulerBackend()
SchedulerBackendImpl
+ executors: List[ExecRef]
+ totalCoreCount: Int
+ requestExecutors()
ExecutorBackend
+ driverUrl: String
+ main()

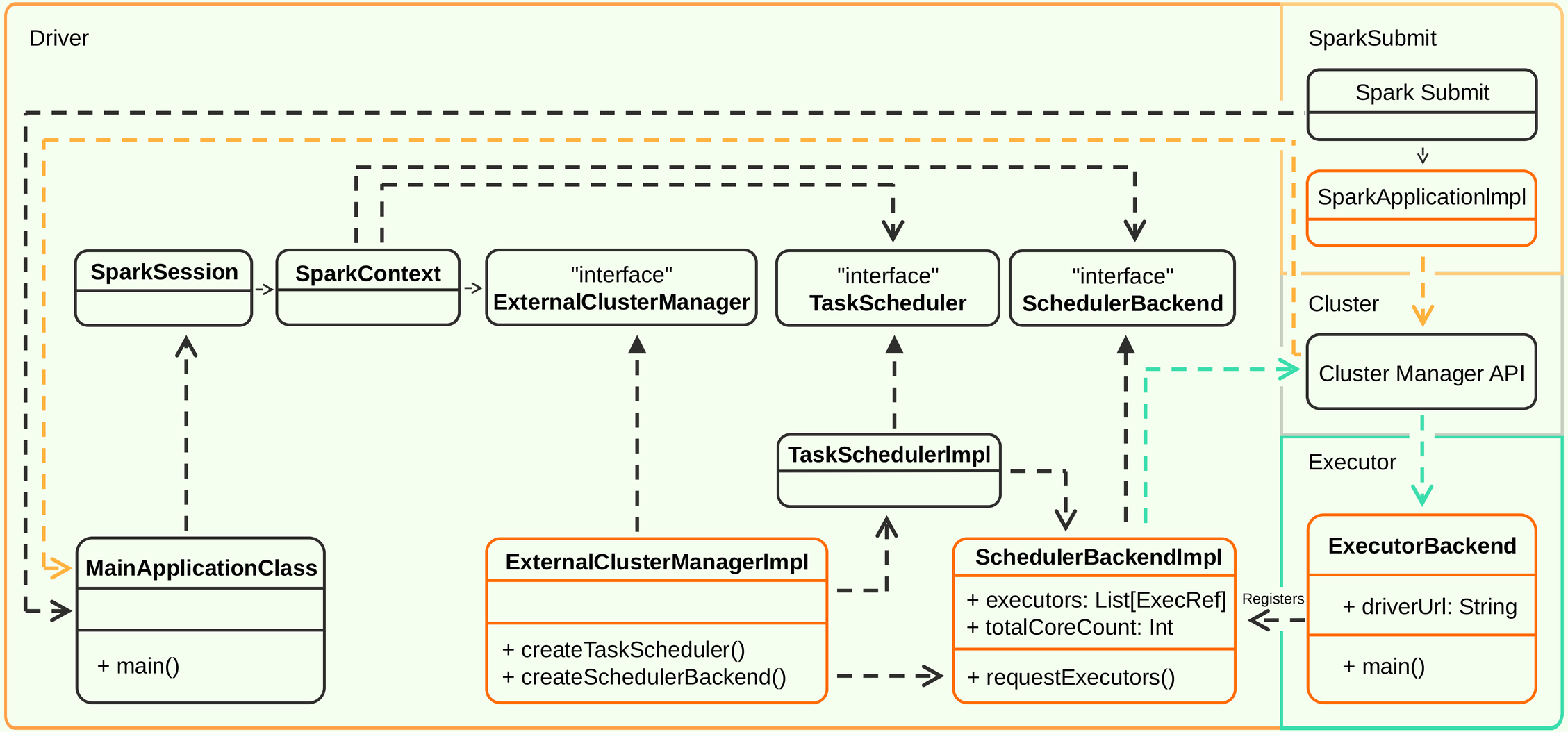

Driver
SparkSubmit
SparkSession
SparkContext
"interface"
ExternalClusterManager
"interface"
TaskScheduler
"interface"
SchedulerBackend
Cluster
Cluster Manager API
TaskSchedulerImpl
Executor
MainApplicationClass
+ main()
ExternalClusterManagerImpl
+ createTaskScheduler()
+ createSchedulerBackend()
SchedulerBackendImpl
+ executors: List[ExecRef]
+ totalCoreCount: Int
+ requestExecutors()
Registers
ExecutorBackend
+ driverUrl: String
+ main()


Driver
SparkSubmit
Spark Submit
SparkSession
SparkContext
"interface"
ExternalClusterManager
"interface"
TaskScheduler
"interface"
SchedulerBackend
Cluster
Cluster Manager API
TaskSchedulerImpl
Executor
MainApplicationClass
+ main()
ExternalClusterManagerImpl
+ createTaskScheduler()
+ createSchedulerBackend()
SchedulerBackendImpl
+ executors: List[ExecRef]
+ totalCoreCount: Int
+ requestExecutors()
Registers
ExecutorBackend
+ driverUrl: String
+ main()


Driver
SparkSubmit
Spark Submit
SparkApplicationImpl
SparkSession
SparkContext
"interface"
ExternalClusterManager
"interface"
TaskScheduler
"interface"
SchedulerBackend
Cluster
Cluster Manager API
TaskSchedulerImpl
Executor
MainApplicationClass
+ main()
ExternalClusterManagerImpl
+ createTaskScheduler()
+ createSchedulerBackend()
SchedulerBackendImpl
+ executors: List[ExecRef]
+ totalCoreCount: Int
+ requestExecutors()
Registers
ExecutorBackend
+ driverUrl: String
+ main()


Driver
SparkSubmit
Spark Submit
SparkApplicationImpl
Cluster
Cluster Manager API
Executor
ExecutorBackend
+ driverUrl: String
+ main()
SparkSession
SparkContext
"interface"
ExternalClusterManager
"interface"
TaskScheduler
"interface"
SchedulerBackend
TaskSchedulerImpl
MainApplicationClass
+ main()
ExternalClusterManagerImpl
+ createTaskScheduler()
+ createSchedulerBackend()
SchedulerBackendImpl
+ executors: List[ExecRef]
+ totalCoreCount: Int
+ requestExecutors()
Registers


Driver
SparkSubmit
Spark Submit
SparkApplicationImpl
Cluster
Cluster Manager API
Executor
SparkSession
SparkContext
"interface"
ExternalClusterManager
"interface"
TaskScheduler
"interface"
SchedulerBackend
TaskSchedulerImpl
MainApplicationClass
+ main()
ExternalClusterManagerImpl
+ createTaskScheduler()
+ createSchedulerBackend()
SchedulerBackendImpl
+ executors: List[ExecRef]
+ totalCoreCount: Int
+ requestExecutors()
Registers
ExecutorBackend
+ driverUrl: String
+ main()


Driver
SparkSubmit
Spark Submit
SparkApplicationImpl
Cluster
Cluster Manager API
Executor
SparkSession
SparkContext
"interface"
ExternalClusterManager
"interface"
TaskScheduler
"interface"
SchedulerBackend
TaskSchedulerImpl
MainApplicationClass
+ main()
ExternalClusterManagerImpl
+ createTaskScheduler()
+ createSchedulerBackend()
SchedulerBackendImpl
+ executors: List[ExecRef]
+ totalCoreCount: Int
+ requestExecutors()
Registers
ExecutorBackend
+ driverUrl: String
+ main()

SPI для Spark resource manager

- ExternalClusterManager

SPI для Spark resource manager

- ExternalClusterManager
- SchedulerBackend

SPI для Spark resource manager

- ExternalClusterManager
- SchedulerBackend
- ExecutorBackend

SPI для Spark resource manager

- ExternalClusterManager
- SchedulerBackend
- ExecutorBackend
- ClusterApplication

SPI для Spark resource manager

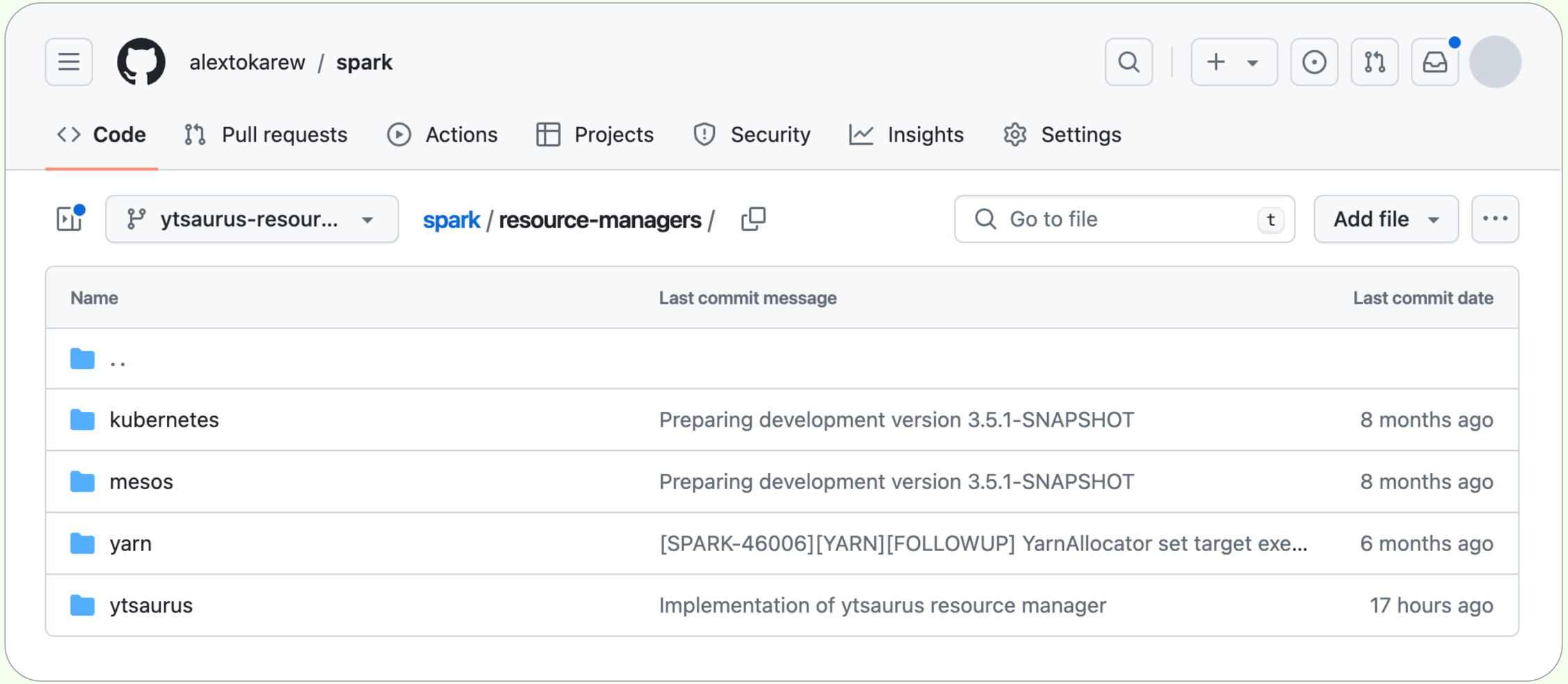
- ExternalClusterManager
- SchedulerBackend
- ExecutorBackend
- ClusterApplication
- _____

SPI для Spark resource manager

- ExternalClusterManager
- SchedulerBackend
- ExecutorBackend
- ClusterApplication
- ———————————
- SparkSubmit — поддержка master url
  и доп. конфигурационных параметров

Реализация YTsaurus resource manager

Cluster managers implementations



YTsaurusClusterManager

spark / core / src / main / scala / org / apache / spark / **SparkContext.scala**

Code | Blame | 3542 lines (3215 loc) · 142 KB

```scala
3328    private def getClusterManager(url: String): Option[ExternalClusterManager] = {
3329      val loader = Utils.getContextOrSparkClassLoader
3330      val serviceLoaders =
3331        ServiceLoader.load(classOf[ExternalClusterManager], loader).asScala.filter(_.canCreate(url))
3332      if (serviceLoaders.size > 1) {
3333        throw new SparkException(
3334          s"Multiple external cluster managers registered for the url $url: $serviceLoaders")
3335      }
3336      serviceLoaders.headOption
3337    }
```

spark / resource-managers / ytsaurus / src / main / resources / META-INF / services / **org.apache.spark.scheduler.ExternalClusterManager**

Code | Blame | 1 lines (1 loc) · 66 Bytes

```
1    org.apache.spark.scheduler.cluster.ytsaurus.YTsaurusClusterManager
```

YTsaurusClusterManager

```scala
private[spark] class YTsaurusClusterManager extends ExternalClusterManager with Logging {

  override def canCreate(masterURL: String): Boolean = {
    masterURL.startsWith("ytsaurus://")
  }

  override def createTaskScheduler(sc: SparkContext, masterURL: String): TaskScheduler = {
    new TaskSchedulerImpl(sc)
  }

  override def createSchedulerBackend(sc: SparkContext, masterURL: String, scheduler: TaskScheduler): SchedulerBackend = {
    // ... Configuration setup ...
    val operationManager = YTsaurusOperationManager.create(ytProxy, sc.conf, networkName)

    new YTsaurusSchedulerBackend(scheduler.asInstanceOf[TaskSchedulerImpl], sc, operationManager)
  }

  override def initialize(scheduler: TaskScheduler, backend: SchedulerBackend): Unit = {
    scheduler.asInstanceOf[TaskSchedulerImpl].initialize(backend)
    backend.asInstanceOf[YTsaurusSchedulerBackend].initialize()
  }
}
```

Фасад для взаимодействия с YTsaurus

- Запуск Vanilla-операции
  с экзекьюторами

Фасад для взаимодействия с YTsaurus

- Запуск Vanilla-операции
  с экзекьюторами
- Запуск Vanilla-операции с драйвером
  (для кластерного режима)

Фасад для взаимодействия с YTsaurus

- Запуск Vanilla-операции
  с экзекьюторами
- Запуск Vanilla-операции с драйвером
  (для кластерного режима)
- Запрос статуса операции драйвера

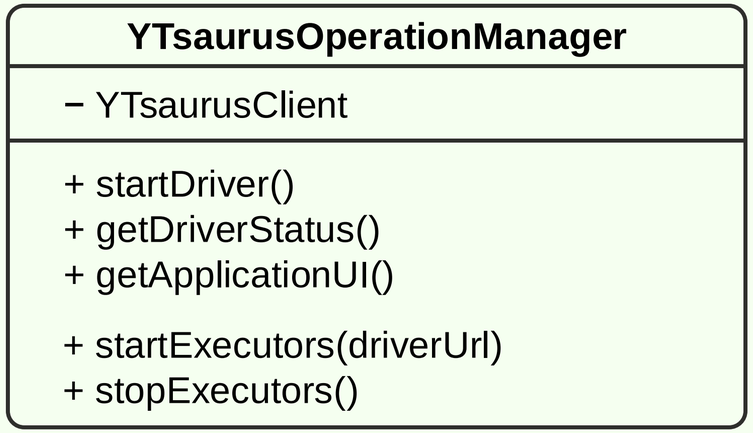
Фасад для взаимодействия с YTsaurus

- Запуск Vanilla-операции
  с экзекьюторами
- Запуск Vanilla-операции с драйвером
  (для кластерного режима)
- Запрос статуса операции драйвера
- Получение URL Application UI-драйвера

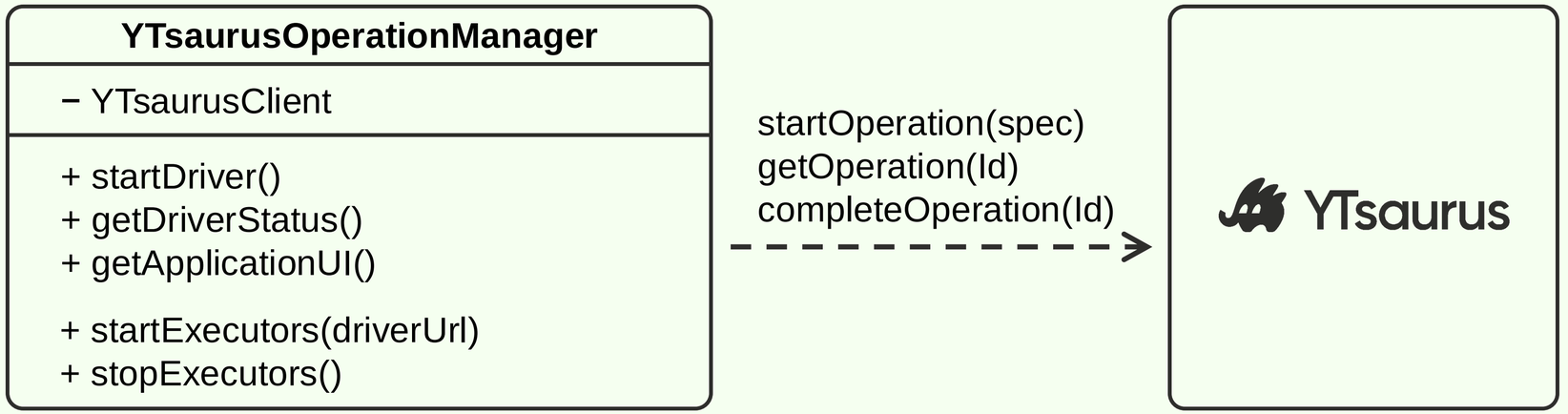
Фасад для взаимодействия с YTsaurus

- Запуск Vanilla-операции
  с экзекьюторами
- Запуск Vanilla-операции с драйвером
  (для кластерного режима)
- Запрос статуса операции драйвера
- Получение URL Application UI-драйвера
- Остановка операции с экзекьюторами
  при закрытии SparkContext

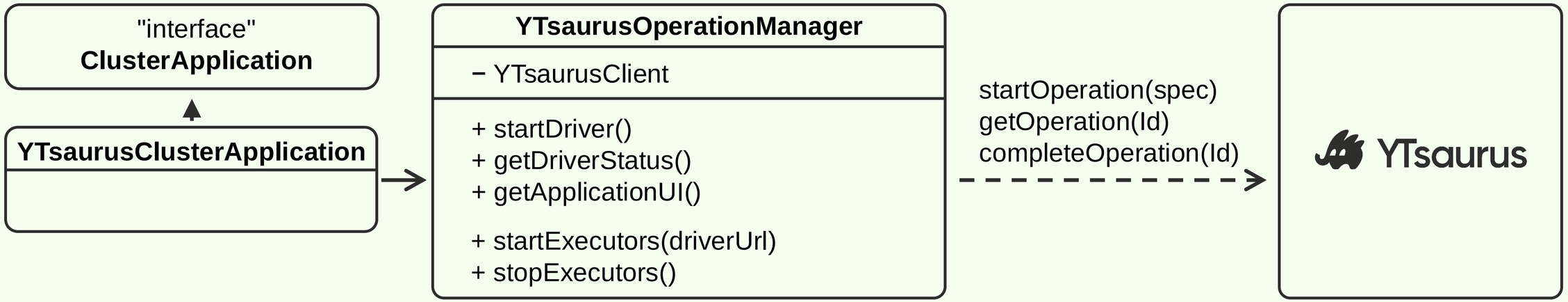
Взаимодействие через YTsaurusOperationManager



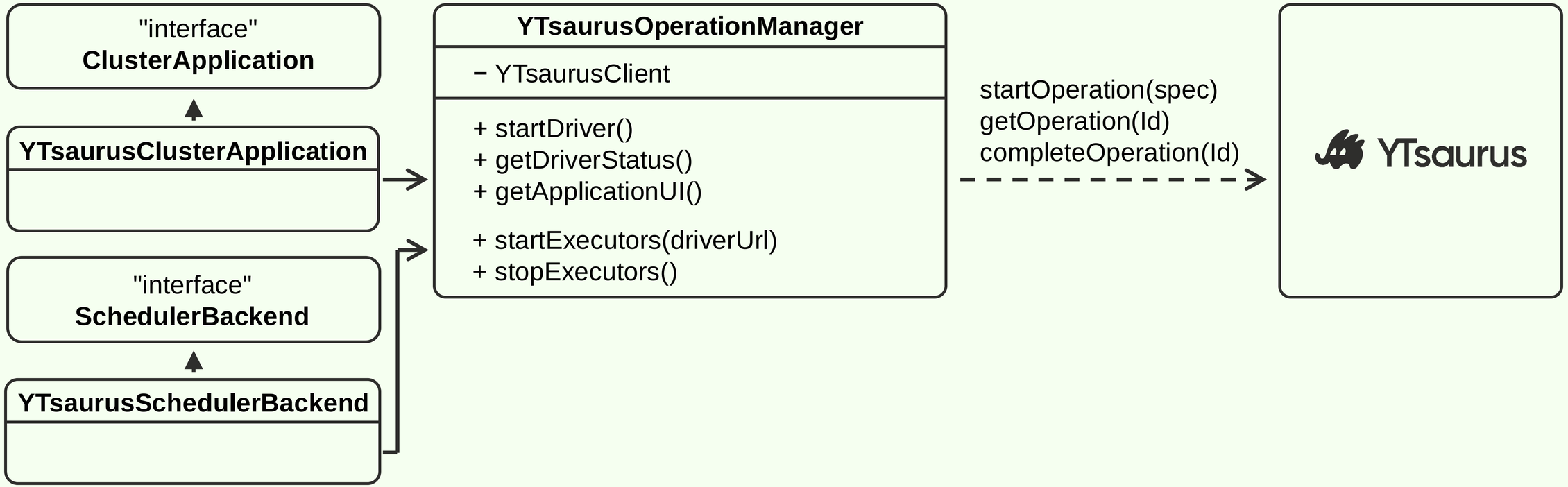
Взаимодействие через YTsaurusOperationManager



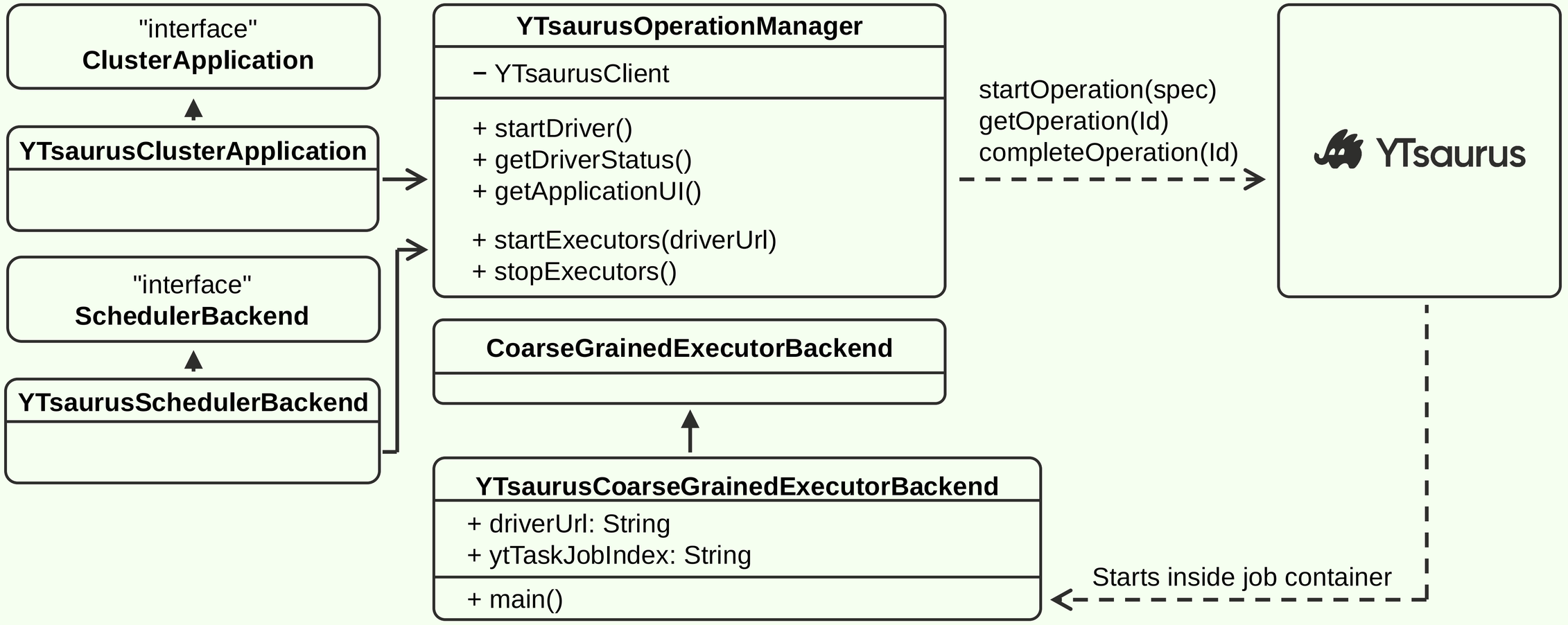
Взаимодействие через YTsaurusOperationManager



Взаимодействие через YTsaurusOperationManager



Взаимодействие через YTsaurusOperationManager



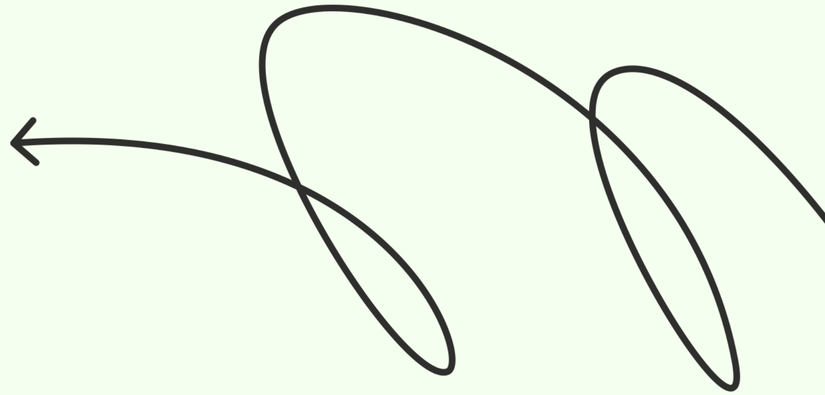
YTsaurusClusterApplication

```scala
private[spark] class YTsaurusClusterApplication extends SparkApplication with Logging {

  override def start(args: Array[String], conf: SparkConf): Unit = {
    // ... Some initial confoguration ...
    val operationManager = YTsaurusOperationManager.create(endpoint, conf)

    try {
      val driverOperation = operationManager.startDriver(conf, appArgs)

      var currentState = "undefined"
      while (!YTsaurusOperationManager.isFinalState(currentState)) {
        Thread.sleep(pingInterval)
        val opSpec = operationManager.getOperation(driverOperation)
        currentState = getOperationState(opSpec)
        logInfo(s"Operation: ${driverOperation.id}, State: $currentState")
        // ...
      }
    } finally {
      operationManager.close()
    }
  }
}
```

YTsaurusSchedulerBackend

```
private[spark] class YTsaurusSchedulerBackend (...) extends CoarseGrainedSchedulerBackend(...) {

  def initialize(): Unit = {
    sc.uiWebUrl.foreach { webUiUrl =>
      sys.env.get("YT_OPERATION_ID").foreach { operationId =>
        operationManager.setOperationDescription(operationId, Map(WEB_UI_KEY -> webUiUrl))
      }
    }
  }

  override def start(): Unit = {
    super.start()
    operationManager.startExecutors(sc, applicationId(), defaultProfile, initialExecutors)
  }

  override def stop(): Unit = {
    super.stop()
    Thread.sleep(conf.get(EXECUTOR_OPERATION_SHUTDOWN_DELAY))
    operationManager.stopExecutors(sc)
    operationManager.close()
  }
  // ...
}
```

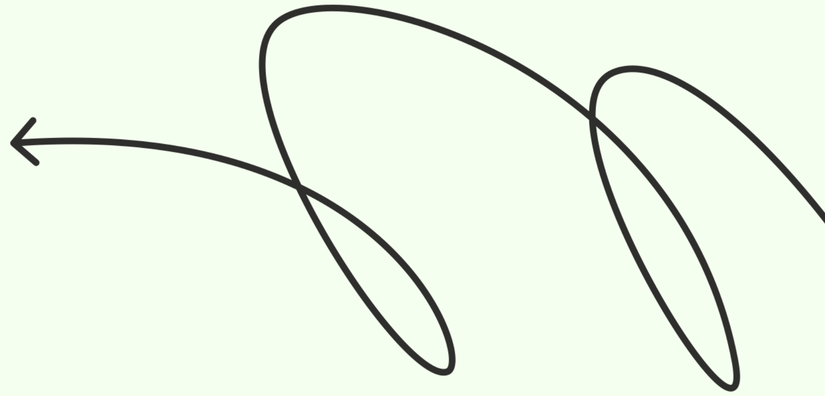
Что нужно добавить в класс SparkSubmit



1

Поддержка
нового паттерна
master URL

Что нужно добавить в класс SparkSubmit



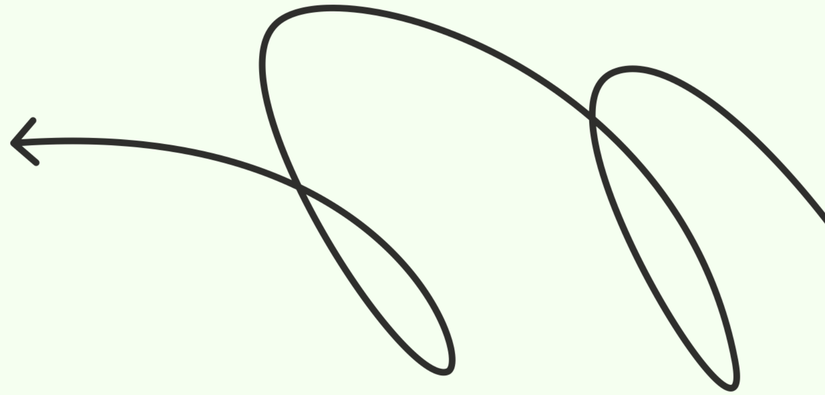
1

Поддержка
нового паттерна
master URL

2

Поддержка
дополнительных
параметров команды
spark-submit

Что нужно добавить в класс SparkSubmit



1

Поддержка нового паттерна master URL

2

Поддержка дополнительных параметров команды spark-submit

3

Вызов класса YTsaurusClusterApplication для запуска в кластерном режиме

Как модифицировать SparkSubmit

Быстрые решения:

Как модифицировать SparkSubmit

Быстрые решения:

- Форк спарка

Как модифицировать SparkSubmit

Быстрые решения:

- Форк спарка
- Рантайм патчинг:

clck.ru/3CGBoY

Как модифицировать SparkSubmit

Быстрые решения:

- Форк спарка
- Рантайм патчинг:

clck.ru/3CGBoY

- Запуск с использованием подкласса SparkSubmit

Как модифицировать SparkSubmit

Быстрые решения:

- Форк спарка
- Рантайм патчинг:

clck.ru/3CGBoY

- Запуск с использованием подкласса SparkSubmit

Правильный путь: интеграция в основной репозиторий Apache Spark:

Как модифицировать SparkSubmit

Быстрые решения:

- Форк спарка
- Рантайм патчинг:

clck.ru/3CGBoY

- Запуск с использованием подкласса SparkSubmit

Правильный путь: интеграция в основной репозиторий Apache Spark:

- Долго по времени

Как модифицировать SparkSubmit

Быстрые решения:

- Форк спарка
- Рантайм патчинг:

clck.ru/3CGBoY

- Запуск с использованием подкласса SparkSubmit

Правильный путь: интеграция в основной репозиторий Apache Spark:

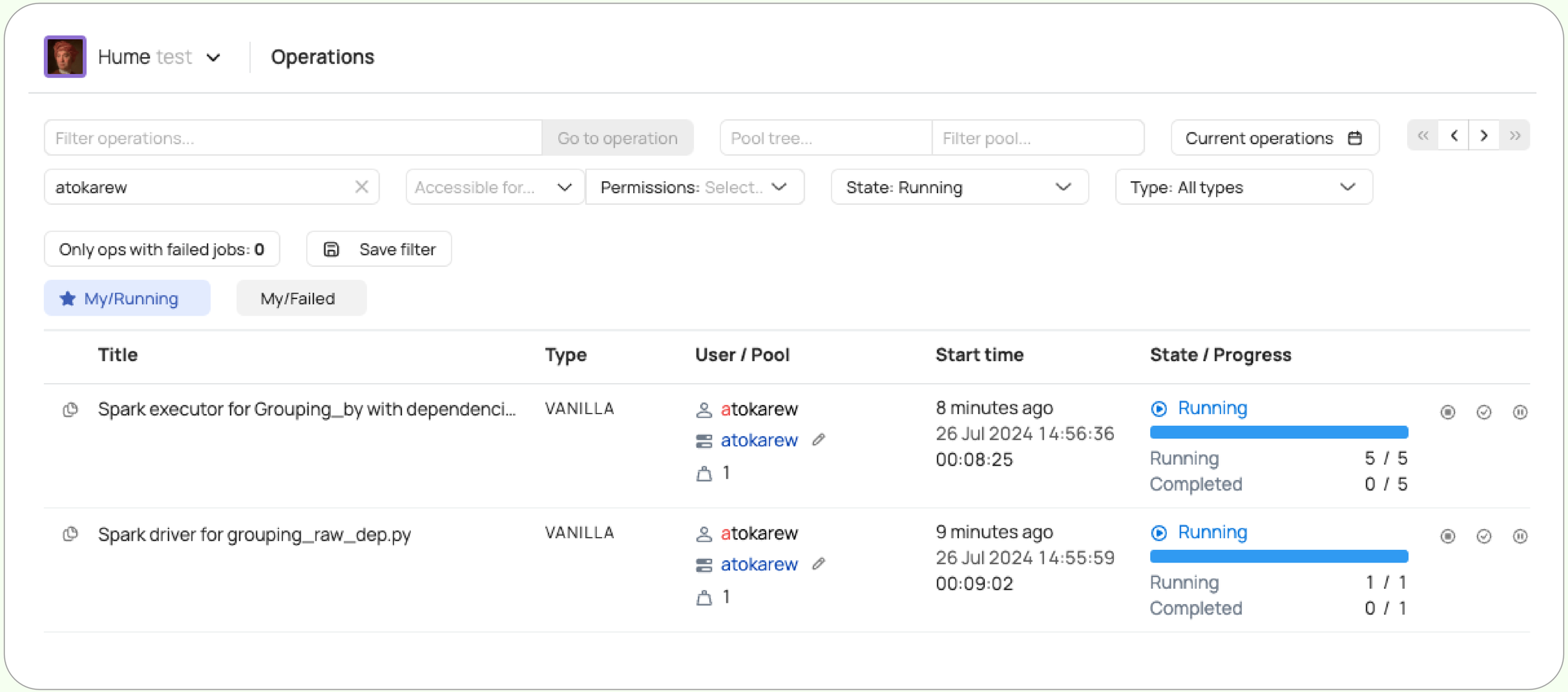
- Долго по времени
- Будет доступен только для новых версий Spark

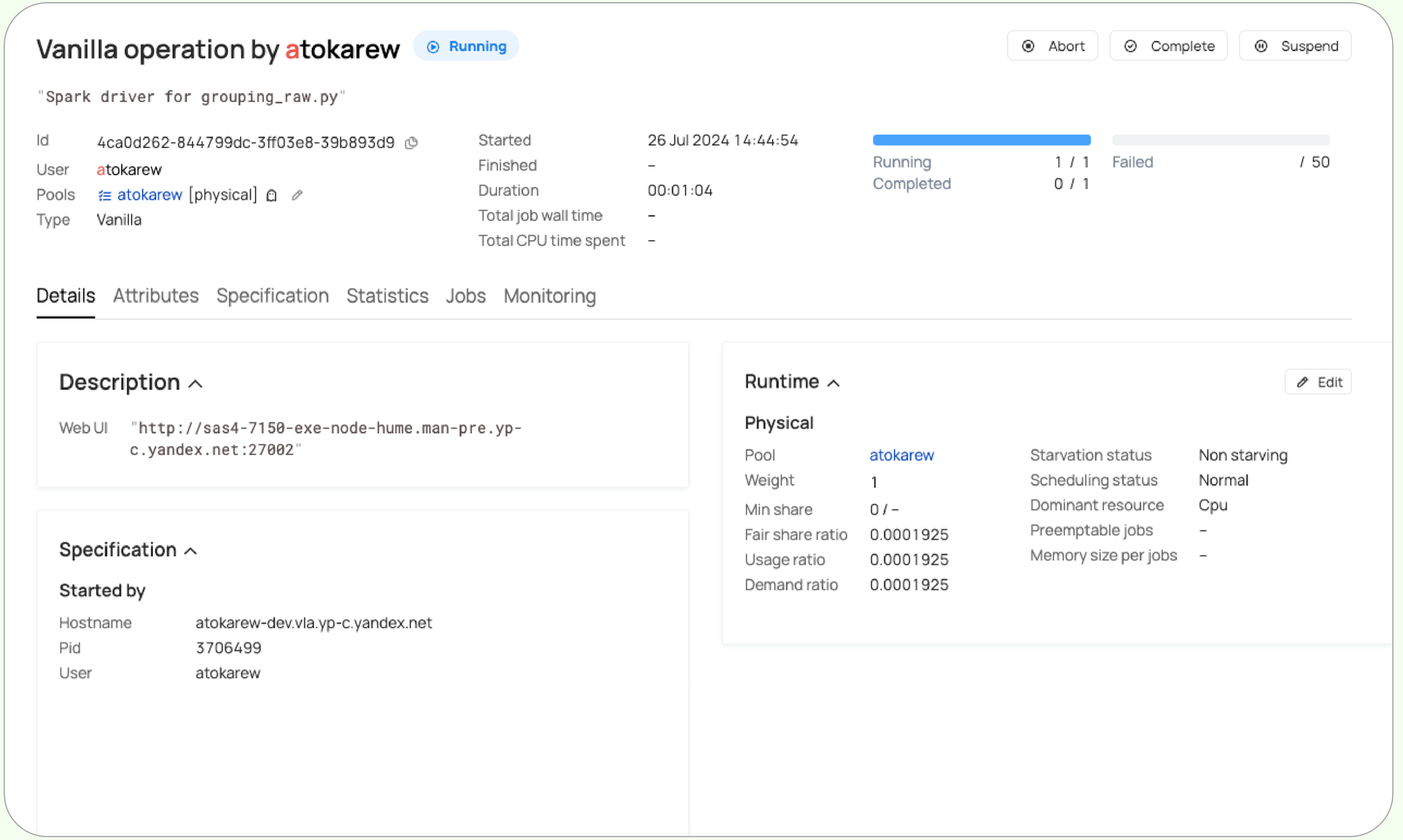
Примеры использования

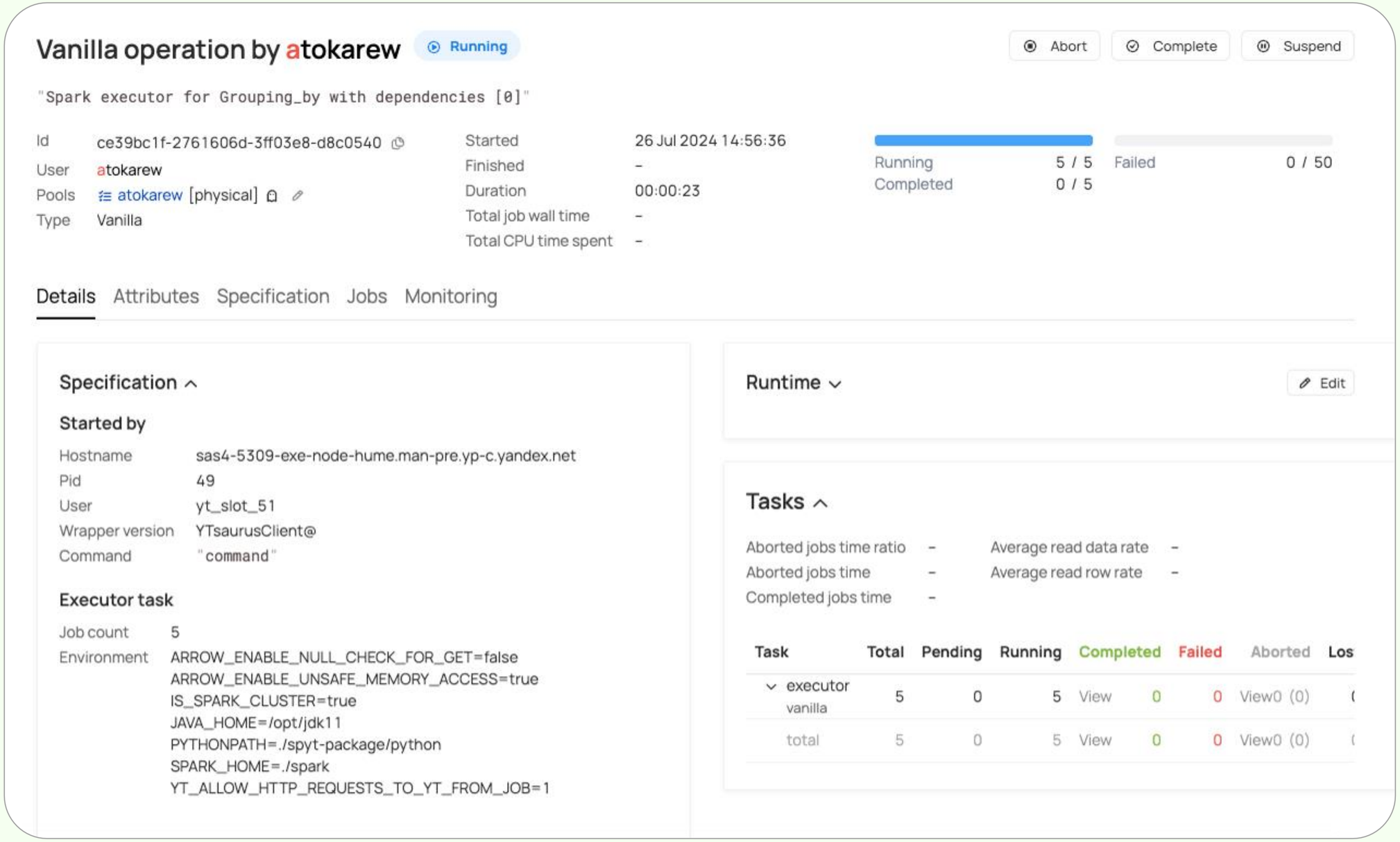
Запуск в кластерном режиме

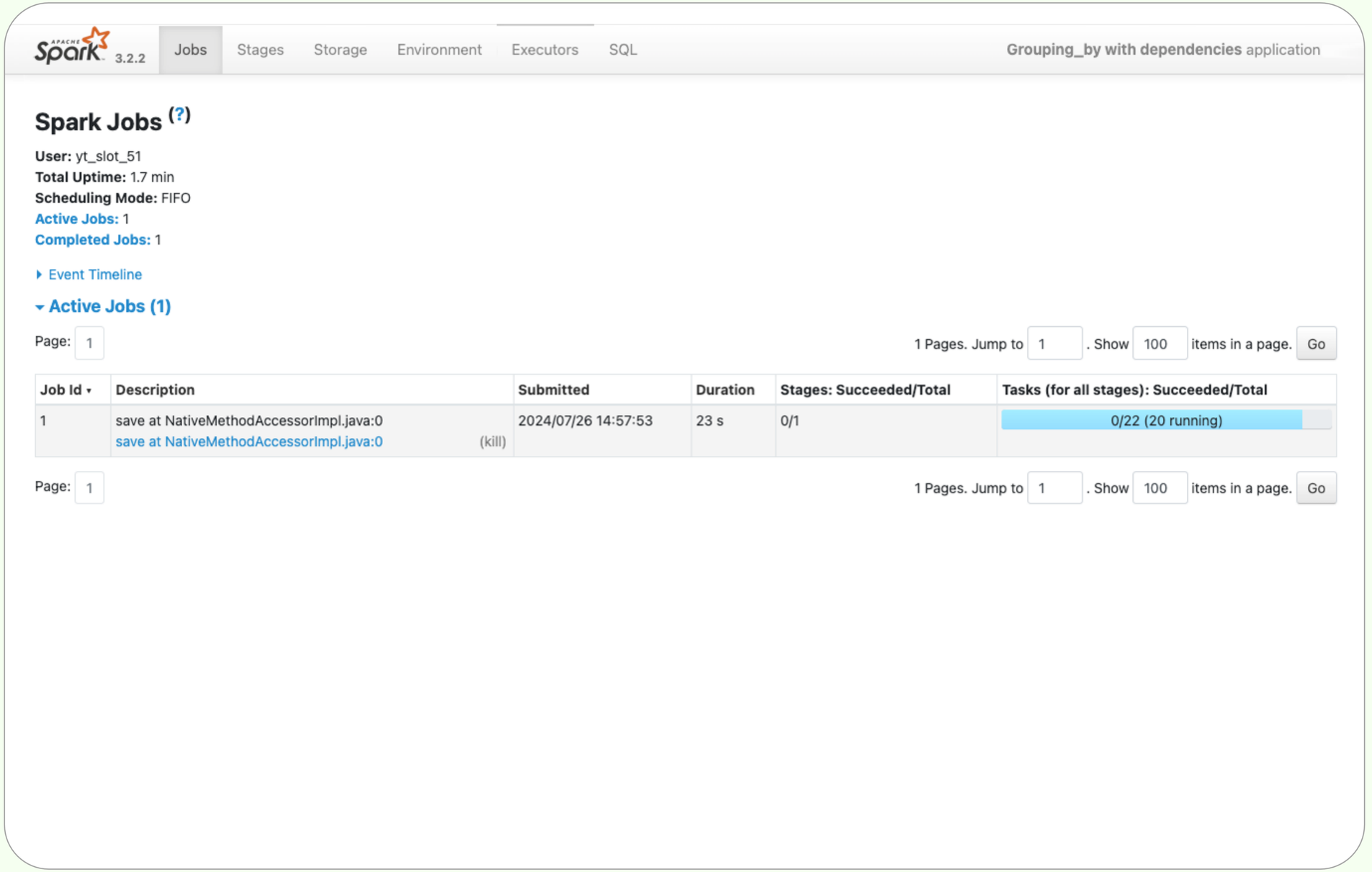
```
spark-submit \
--master ytsaurus://my.ytsaurus.company.net \
--deploy-mode cluster \
--queue atokarew \
--num-executors 5 \
--executor-cores 4 \
--py-files yt:///home/atokarew/scripts/deps.py \
yt:///home/atokarew/scripts/grouping_raw_dep.py
```

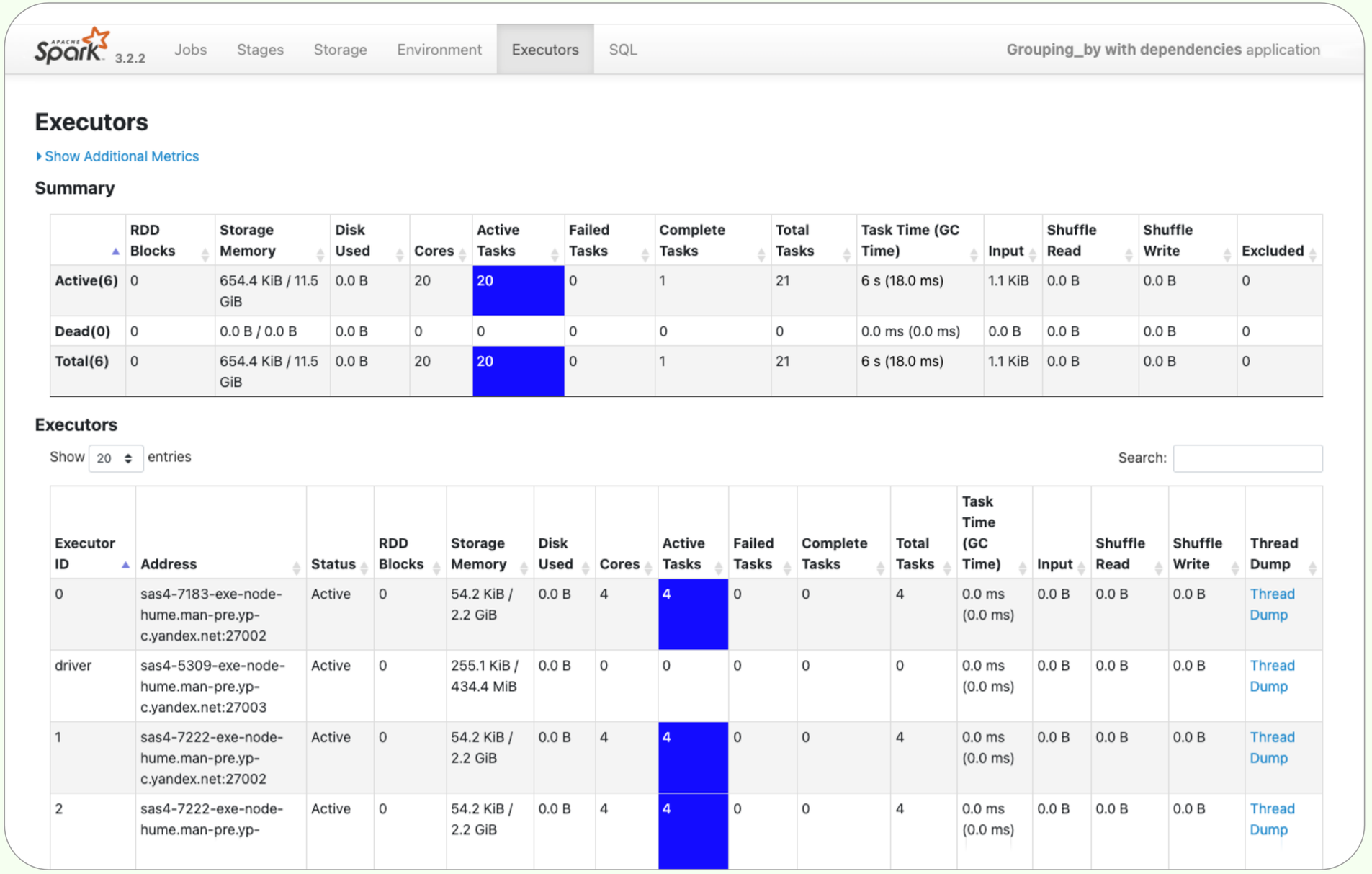
```
24/07/26 14:36:53 INFO YTsaurusClusterApplication: Operation: 1b943a93-a98e1b7f-3ff03e8-93890bc, State: running
24/07/26 14:36:53 INFO YTsaurusClusterApplication: Web UI: http://sas4-5356-exe-node-hume.man-pre.yp-c.yandex.net:27002
24/07/26 14:36:56 INFO YTsaurusClusterApplication: Operation: 1b943a93-a98e1b7f-3ff03e8-93890bc, State: running
24/07/26 14:36:59 INFO YTsaurusClusterApplication: Operation: 1b943a93-a98e1b7f-3ff03e8-93890bc, State: running
24/07/26 14:37:03 INFO YTsaurusClusterApplication: Operation: 1b943a93-a98e1b7f-3ff03e8-93890bc, State: running
24/07/26 14:37:06 INFO YTsaurusClusterApplication: Operation: 1b943a93-a98e1b7f-3ff03e8-93890bc, State: running
24/07/26 14:37:09 INFO YTsaurusClusterApplication: Operation: 1b943a93-a98e1b7f-3ff03e8-93890bc, State: running
24/07/26 14:37:12 INFO YTsaurusClusterApplication: Operation: 1b943a93-a98e1b7f-3ff03e8-93890bc, State: running
24/07/26 14:37:15 INFO YTsaurusClusterApplication: Operation: 1b943a93-a98e1b7f-3ff03e8-93890bc, State: running
24/07/26 14:37:18 INFO YTsaurusClusterApplication: Operation: 1b943a93-a98e1b7f-3ff03e8-93890bc, State: running
24/07/26 14:37:21 INFO YTsaurusClusterApplication: Operation: 1b943a93-a98e1b7f-3ff03e8-93890bc, State: running
24/07/26 14:37:24 INFO YTsaurusClusterApplication: Operation: 1b943a93-a98e1b7f-3ff03e8-93890bc, State: running
24/07/26 14:37:27 INFO YTsaurusClusterApplication: Operation: 1b943a93-a98e1b7f-3ff03e8-93890bc, State: running
24/07/26 14:37:30 INFO YTsaurusClusterApplication: Operation: 1b943a93-a98e1b7f-3ff03e8-93890bc, State: running
24/07/26 14:37:34 INFO YTsaurusClusterApplication: Operation: 1b943a93-a98e1b7f-3ff03e8-93890bc, State: running
24/07/26 14:37:37 INFO YTsaurusClusterApplication: Operation: 1b943a93-a98e1b7f-3ff03e8-93890bc, State: running
24/07/26 14:37:40 INFO YTsaurusClusterApplication: Operation: 1b943a93-a98e1b7f-3ff03e8-93890bc, State: running
24/07/26 14:37:43 INFO YTsaurusClusterApplication: Operation: 1b943a93-a98e1b7f-3ff03e8-93890bc, State: running
24/07/26 14:37:46 INFO YTsaurusClusterApplication: Operation: 1b943a93-a98e1b7f-3ff03e8-93890bc, State: completed
```


Hume test
Operations
Filter operations...
Go to operation
Pool tree...
Filter pool...
Current operations
atokarew
Accessible for...
Permissions: Select..
State: Running
Type: All types
Only ops with failed jobs: 0
Save filter
My/Running
My/Failed
Title
Type
User / Pool
Start time
State / Progress
Spark executor for Grouping_by with dependenci...
VANILLA
atokarew
atokarew
1
8 minutes ago
26 Jul 2024 14:56:36
00:08:25
Running
Running 5 / 5
Completed 0 / 5
Spark driver for grouping_raw_dep.py
VANILLA
atokarew
atokarew
1
9 minutes ago
26 Jul 2024 14:55:59
00:09:02
Running
Running 1 / 1
Completed 0 / 1


Vanilla operation by atokarew
Running
Abort
Complete
Suspend
"Spark driver for grouping_raw.py"
Id 4ca0d262-844799dc-3ff03e8-39b893d9
User atokarew
Pools atokarew [physical]
Type Vanilla
Started 26 Jul 2024 14:44:54
Finished –
Duration 00:01:04
Total job wall time –
Total CPU time spent –
Running 1 / 1
Completed 0 / 1
Failed / 50
Details Attributes Specification Statistics Jobs Monitoring
Description
Web UI "http://sas4-7150-exe-node-hume.man-pre.yp-c.yandex.net:27002"
Specification
Started by
Hostname atokarew-dev.vla.yp-c.yandex.net
Pid 3706499
User atokarew
Runtime
Edit
Physical
Pool atokarew
Weight 1
Min share 0 / –
Fair share ratio 0.0001925
Usage ratio 0.0001925
Demand ratio 0.0001925
Starvation status Non starving
Scheduling status Normal
Dominant resource Cpu
Preemptable jobs –
Memory size per jobs –


Vanilla operation by atokarew
Running
Abort
Complete
Suspend
"Spark executor for Grouping_by with dependencies [0]"
Id ce39bc1f-2761606d-3ff03e8-d8c0540
User atokarew
Pools atokarew [physical]
Type Vanilla
Started 26 Jul 2024 14:56:36
Finished -
Duration 00:00:23
Total job wall time -
Total CPU time spent -
Running 5 / 5
Completed 0 / 5
Failed 0 / 50
Details
Attributes
Specification
Jobs
Monitoring
Specification
Started by
Hostname sas4-5309-exe-node-hume.man-pre.yp-c.yandex.net
Pid 49
User yt_slot_51
Wrapper version YTsaurusClient@
Command "command"
Executor task
Job count 5
Environment ARROW_ENABLE_NULL_CHECK_FOR_GET=false
ARROW_ENABLE_UNSAFE_MEMORY_ACCESS=true
IS_SPARK_CLUSTER=true
JAVA_HOME=/opt/jdk11
PYTHONPATH=./spyt-package/python
SPARK_HOME=./spark
YT_ALLOW_HTTP_REQUESTS_TO_YT_FROM_JOB=1
Runtime
Edit
Tasks
Aborted jobs time ratio -
Aborted jobs time -
Completed jobs time -
Average read data rate -
Average read row rate -
Task Total Pending Running Completed Failed Aborted Los
executor vanilla 5 0 5 View 0 0 View0 (0)
total 5 0 5 View 0 0 View0 (0)

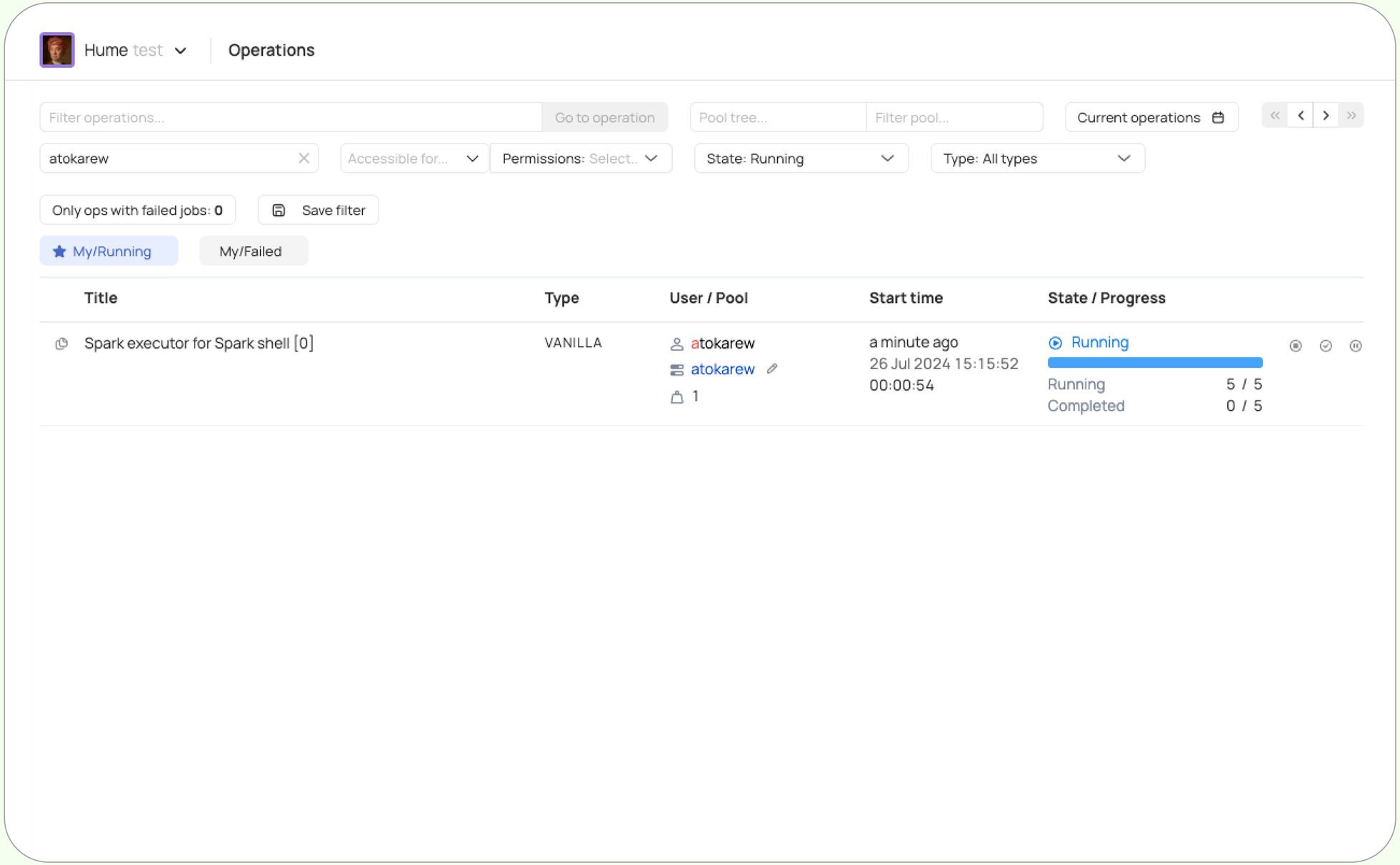

Spark 3.2.2
Jobs
Stages
Storage
Environment
Executors
SQL
Grouping_by with dependencies application
Spark Jobs (?)
User: yt_slot_51
Total Uptime: 1.7 min
Scheduling Mode: FIFO
Active Jobs: 1
Completed Jobs: 1
Event Timeline
Active Jobs (1)
Page: 1
1 Pages. Jump to 1 . Show 100 items in a page. Go
Job Id
Description
Submitted
Duration
Stages: Succeeded/Total
Tasks (for all stages): Succeeded/Total
1
save at NativeMethodAccessorImpl.java:0
save at NativeMethodAccessorImpl.java:0 (kill)
2024/07/26 14:57:53
23 s
0/1
0/22 (20 running)
Page: 1
1 Pages. Jump to 1 . Show 100 items in a page. Go

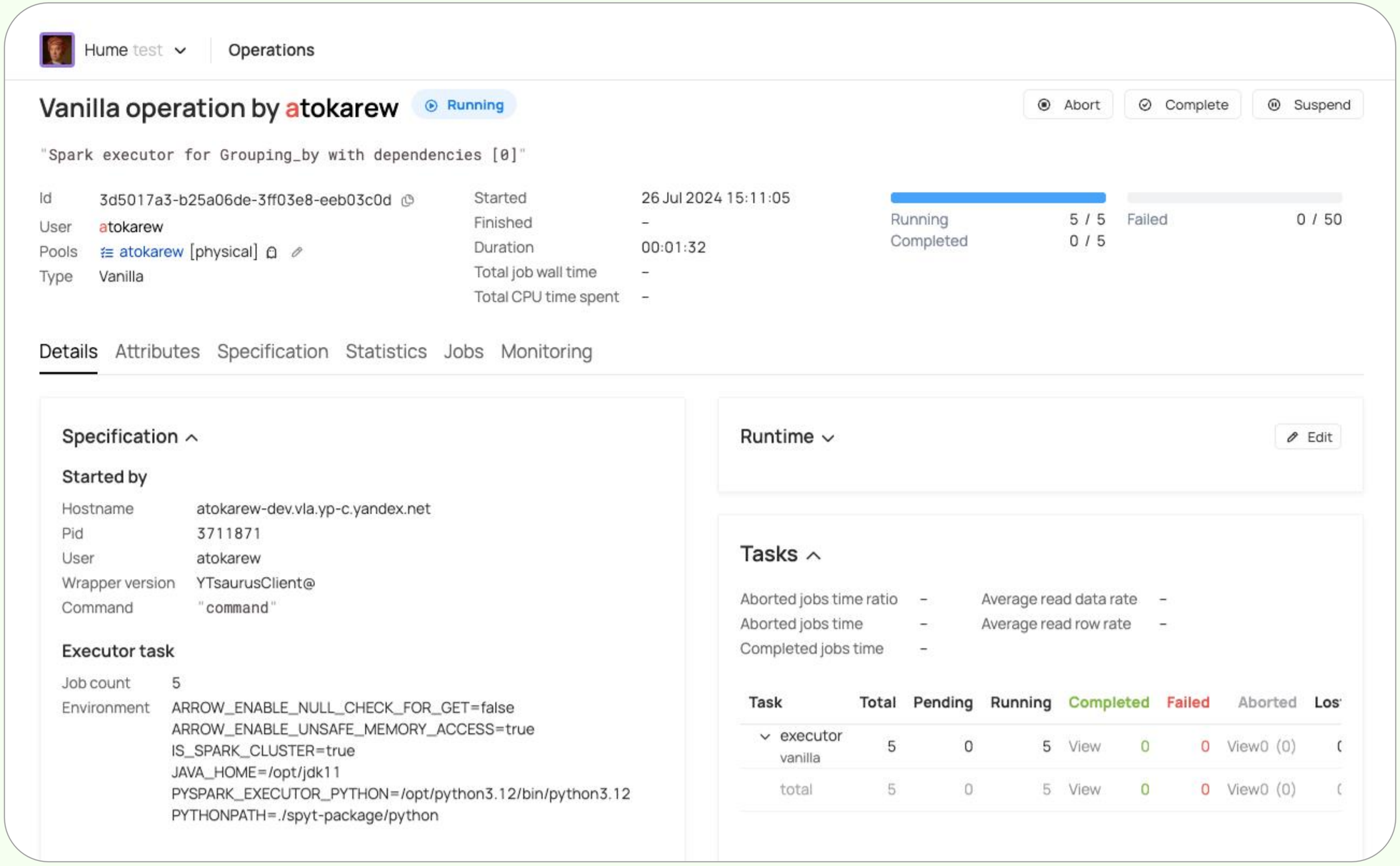

Spark 3.2.2
Jobs
Stages
Storage
Environment
Executors
SQL
Grouping_by with dependencies application
Executors
Show Additional Metrics
Summary
RDD Blocks
Storage Memory
Disk Used
Cores
Active Tasks
Failed Tasks
Complete Tasks
Total Tasks
Task Time (GC Time)
Input
Shuffle Read
Shuffle Write
Excluded
Active(6) 0 654.4 KiB / 11.5 GiB 0.0 B 20 20 0 1 21 6 s (18.0 ms) 1.1 KiB 0.0 B 0.0 B 0
Dead(0) 0 0.0 B / 0.0 B 0.0 B 0 0 0 0 0 0.0 ms (0.0 ms) 0.0 B 0.0 B 0.0 B 0
Total(6) 0 654.4 KiB / 11.5 GiB 0.0 B 20 20 0 1 21 6 s (18.0 ms) 1.1 KiB 0.0 B 0.0 B 0
Executors
Show 20 entries
Search:
Executor ID
Address
Status
RDD Blocks
Storage Memory
Disk Used
Cores
Active Tasks
Failed Tasks
Complete Tasks
Total Tasks
Task Time (GC Time)
Input
Shuffle Read
Shuffle Write
Thread Dump
0 sas4-7183-exe-node-hume.man-pre.yp-c.yandex.net:27002 Active 0 54.2 KiB / 2.2 GiB 0.0 B 4 4 0 0 4 0.0 ms (0.0 ms) 0.0 B 0.0 B 0.0 B Thread Dump
driver sas4-5309-exe-node-hume.man-pre.yp-c.yandex.net:27003 Active 0 255.1 KiB / 434.4 MiB 0.0 B 0 0 0 0 0 0.0 ms (0.0 ms) 0.0 B 0.0 B 0.0 B Thread Dump
1 sas4-7222-exe-node-hume.man-pre.yp-c.yandex.net:27002 Active 0 54.2 KiB / 2.2 GiB 0.0 B 4 4 0 0 4 0.0 ms (0.0 ms) 0.0 B 0.0 B 0.0 B Thread Dump
2 sas4-7222-exe-node-hume.man-pre.yp- Active 0 54.2 KiB / 2.2 GiB 0.0 B 4 4 0 0 4 0.0 ms (0.0 ms) 0.0 B 0.0 B 0.0 B Thread Dump

Запуск spark-shell

```
spark-shell \

--master ytsaurus://my.ytsaurus.company.net \

--num-executors 5 \

--queue atokarew
```

```
(spyt2.1.0) atokarew@atokarew-dev:~$ spark-shell --master ytsaurus://hume.yt.yandex.net --num-executors 5 --queue atokarew
Spark context Web UI available at http://atokarew-dev.vla.yp-c.yandex.net:27002
Spark context available as 'sc' (master = ytsaurus://hume.yt.yandex.net, app id = spark-application-1721996322029).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 3.2.2
      /_/

Using Scala version 2.12.15 (OpenJDK 64-Bit Server VM, Java 11.0.20)
Type in expressions to have them evaluated.
Type :help for more information.

scala> import tech.ytsaurus.spyt._
import tech.ytsaurus.spyt._

scala> val df = spark.read.option("recursiveFileLookup", "false").yt("//home/dev/atokarew/tables/square_roots")
df: org.apache.spark.sql.DataFrame = [id: bigint, sqrt: decimal(7,2)]

scala> df.show(10, false)
+---+----+
|id |sqrt|
+---+----+
|1  |1.00|
|2  |1.41|
|3  |1.73|
|4  |2.00|
|5  |2.24|
|6  |2.45|
```


Hume test
Operations
Filter operations...
Go to operation
Pool tree...
Filter pool...
Current operations
atokarew
Accessible for...
Permissions: Select...
State: Running
Type: All types
Only ops with failed jobs: 0
Save filter
My/Running
My/Failed
Title
Type
User / Pool
Start time
State / Progress
Spark executor for Spark shell [0]
VANILLA
atokarew
atokarew
1
a minute ago
26 Jul 2024 15:15:52
00:00:54
Running
Running 5 / 5
Completed 0 / 5


Hume test
Operations
Vanilla operation by atokarew
Running
Abort
Complete
Suspend
"Spark executor for Grouping_by with dependencies [0]"
Id 3d5017a3-b25a06de-3ff03e8-eeb03c0d
User atokarew
Pools atokarew [physical]
Type Vanilla
Started 26 Jul 2024 15:11:05
Finished -
Duration 00:01:32
Total job wall time -
Total CPU time spent -
Running 5 / 5
Completed 0 / 5
Failed 0 / 50
Details
Attributes
Specification
Statistics
Jobs
Monitoring
Specification
Started by
Hostname atokarew-dev.vla.yp-c.yandex.net
Pid 3711871
User atokarew
Wrapper version YTsaurusClient@
Command "command"
Executor task
Job count 5
Environment ARROW_ENABLE_NULL_CHECK_FOR_GET=false
ARROW_ENABLE_UNSAFE_MEMORY_ACCESS=true
IS_SPARK_CLUSTER=true
JAVA_HOME=/opt/jdk11
PYSPARK_EXECUTOR_PYTHON=/opt/python3.12/bin/python3.12
PYTHONPATH=./spyt-package/python
Runtime
Edit
Tasks
Aborted jobs time ratio -
Aborted jobs time -
Completed jobs time -
Average read data rate -
Average read row rate -
Task Total Pending Running Completed Failed Aborted
executor vanilla 5 0 5 View 0 0 View0 (0)
total 5 0 5 View 0 0 View0 (0)

Что ещё может делать Cluster Manager

Поддержка Dynamic Allocation

- Необходим метод в API YTsaurus, позволяющий изменять количество джобов у работающей операции

Поддержка Dynamic Allocation

- Необходим метод в API YTsaurus, позволяющий изменять количество джобов у работающей операции
- Необходим внешний Shuffle-сервис

Поддержка Dynamic Allocation

- Необходим метод в API YTsaurus, позволяющий изменять количество джобов у работающей операции
- Необходим внешний Shuffle-сервис
- Будет доступно в новом релизе YTsaurus

Dynamic Allocation — альтернативный подход

- Запуск дополнительных операций
  на определенное количество executor-ов

Dynamic Allocation — альтернативный подход

- Запуск дополнительных операций на определенное количество executor-ов
- Недостатки:

Dynamic Allocation — альтернативный подход

- Запуск дополнительных операций на определенное количество executor-ов
- Недостатки:

  Сложно определить, когда можно останавливать доп. операцию

Dynamic Allocation — альтернативный подход

- Запуск дополнительных операций на определенное количество executor-ов
- Недостатки:

  Сложно определить, когда можно останавливать доп. операцию

  Перерасход квоты планировщика на запущенные операции

Поддержка различных профилей ресурсов

- Профили ресурсов позволяет запускать задачи на неоднородных кластерах

Поддержка различных профилей ресурсов

- Профили ресурсов позволяет запускать задачи на неоднородных кластерах
- Используется для запуска задач на GPU

Поддержка различных профилей ресурсов

- Профили ресурсов позволяет запускать задачи на неоднородных кластерах
- Используется для запуска задач на GPU
- Cluster Manager должен уметь запускать executor-ы в соответствии с требуемым запросом на ресурсы

Хранение истории запуска приложений

- Event логи записываются в динамические таблицы YTsaurus

Хранение истории запуска приложений

- Event логи записываются в динамические таблицы YTsaurus
- Динамические таблицы — транзакционный key-value storage

Хранение истории запуска приложений

- Event логи записываются в динамические таблицы YTsaurus
- Динамические таблицы — транзакционный key-value storage
- Две таблицы: сами логи и метаданные

Хранение истории запуска приложений

- Event логи записываются в динамические таблицы YTsaurus
- Динамические таблицы — транзакционный key-value storage
- Две таблицы: сами логи и метаданные
- Для History Server реализован адаптер, позволяющий читать event-логи из этих таблиц

Ссылки

Мы на GitHub:
clck.ru/3CGFsQ

Реализация
resource manager:
clck.ru/3CGG9G

Опенсорс-чат:
t.me/ytsaurus_ru

YTsaurus

Вопросы?

Александр Токарев, Яндекс,
TeamLead группы разработки SPYT
powered by Apache Spark