Hadoop MapReduce

Механизм пакетной обработки ОЧЕНЬ большого объема данных.

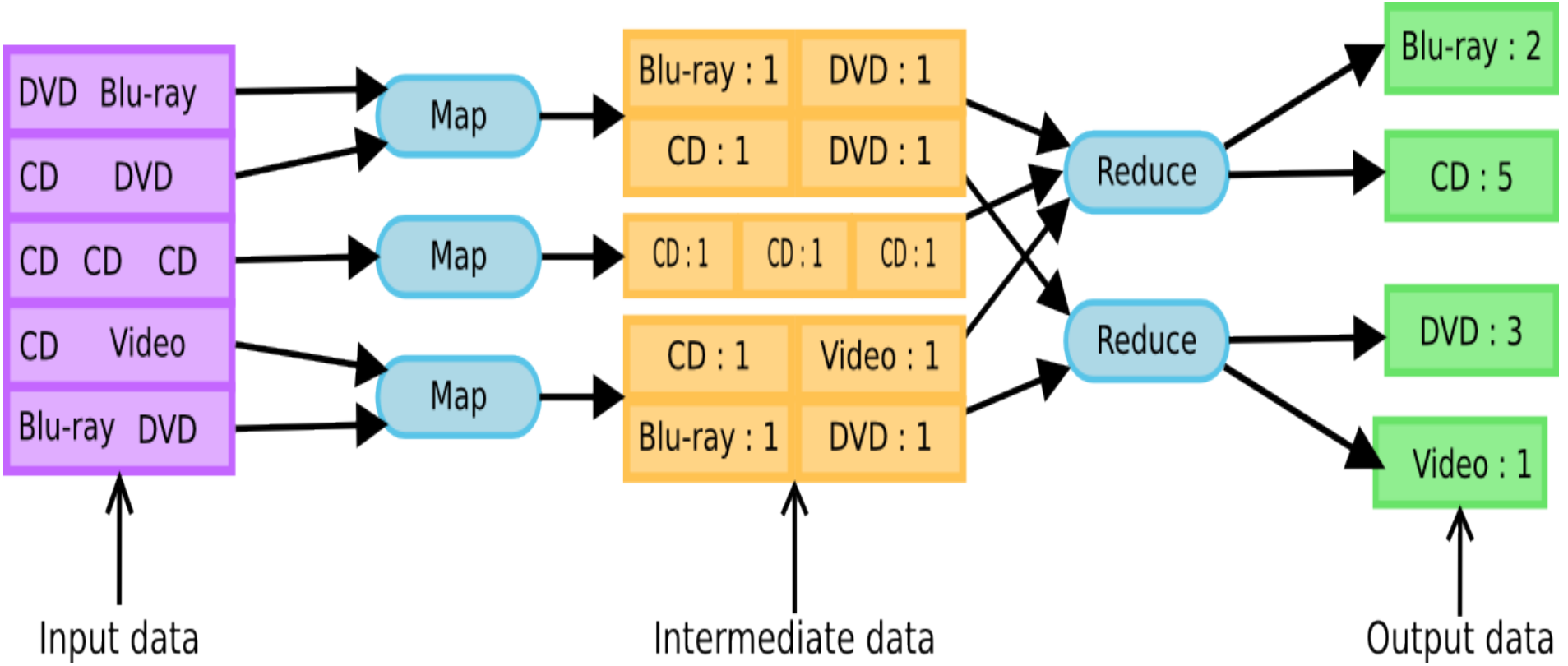
Зачем нужен MapReduce ?

- Параллельная обработка больших объемов данных
- Простые алгоритмы.
- Масштабируемость.

Идея MapReduce

- Программист пишет две функции – map и reduce
- Map применяется к каждой строке распределенного файла (или записи big table) и преобразует ее в набор пар key:value
- Reduce применяется к каждому набору данных key:[value1, value2] и преобразует ее в набор key2:value3

Принципиальная схема



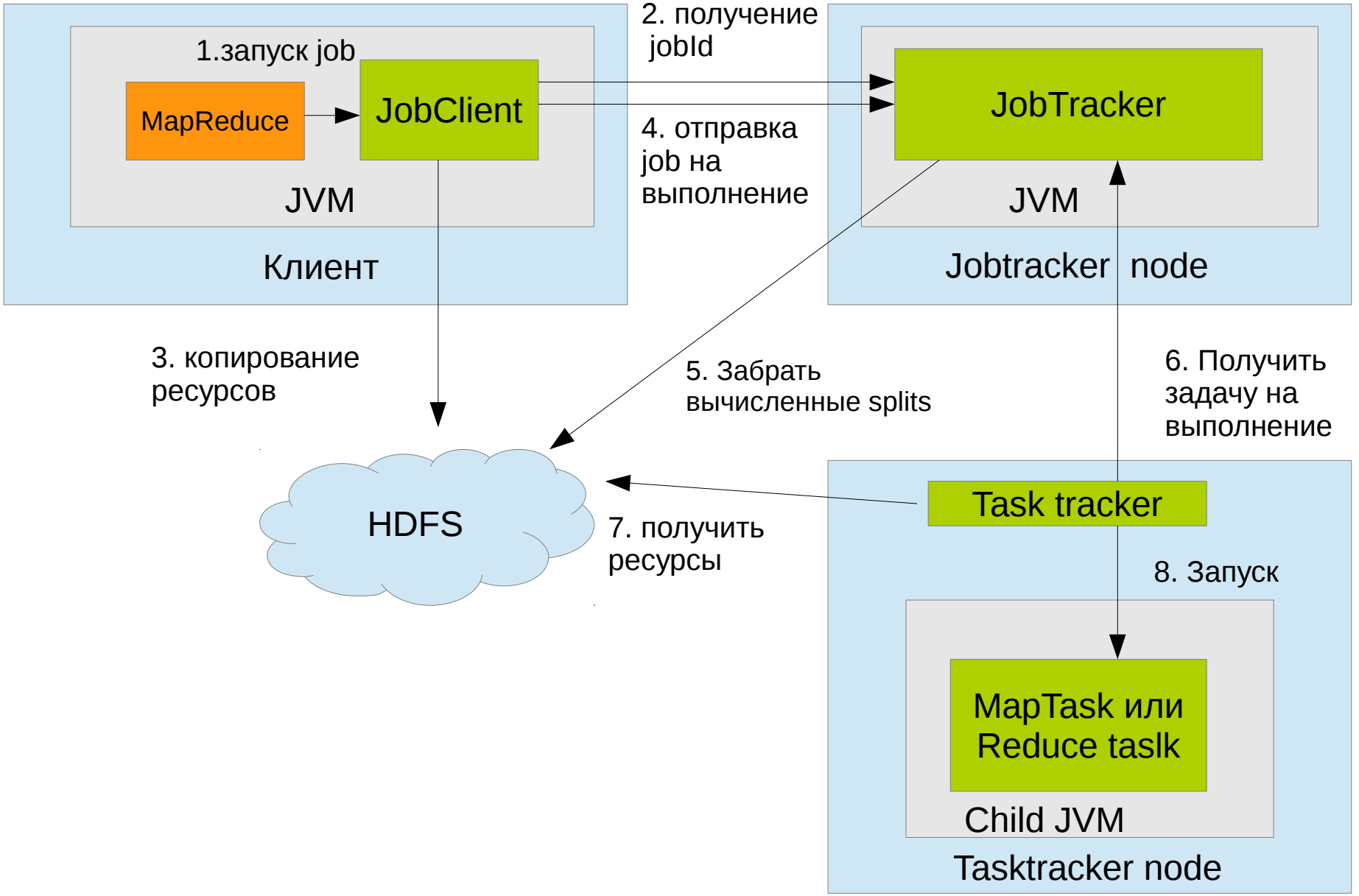
Реализация в Hadoop.Терминология

- Job — это экземпляр процесса обработки данных
- Job состоит из входных данных, функций Map и Reduce и конфигурационной информации
- Hadoop запускает job разделяя процесс выполнения на tasks — задачи
- Tasks — бывают двух видов Map Tasks и Reduce tasks
- Tasks запускаются на узлах hadoop
- После Map возможен также запуск Combiner — который проведет предварительное комбинирование данных локально.

Терминология.Продолжение.

- Узлы бывают двух видов — Tasktrackers и Jobtrackers
- Jobtracker координирует работу Tasktrackers запуская на них задачи (tasks)
- Hadoop разделяет исходные данные на участки (splits) и создает по одному map task на каждый split

Запуск задачи (job)



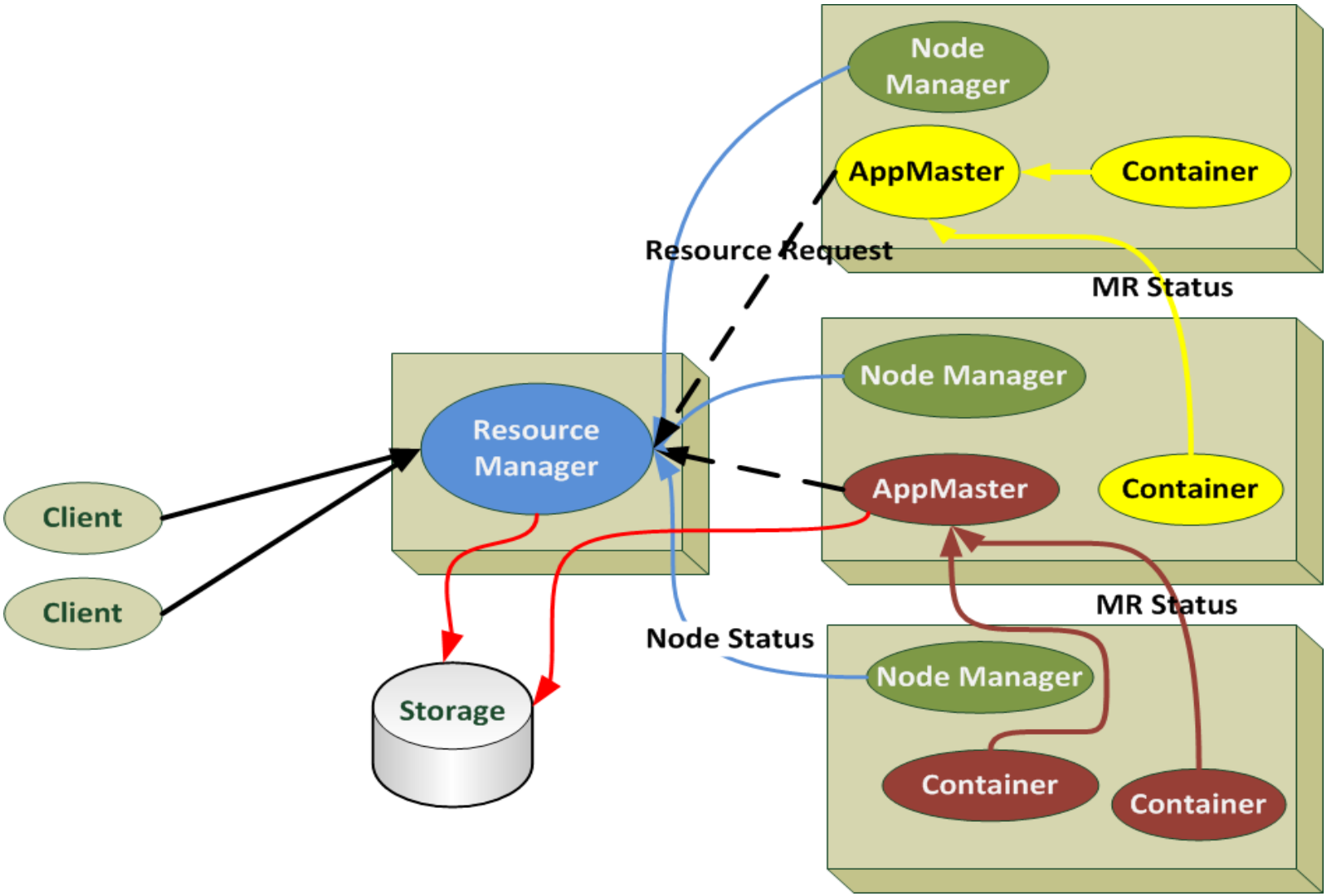
Hadoop 2

- Зачем нам hadoop 2 ?
  - весь кластер hadoop 1 обслуживает только один Job Tracker
    - Есть предел нагрузки
    - В случае падения job tracker — теряются все задачи
  - Для ряда задач не нужен Map Reduce, но есть желание использовать инфраструктуру hadoop для централизованого запуска задач и управления ими

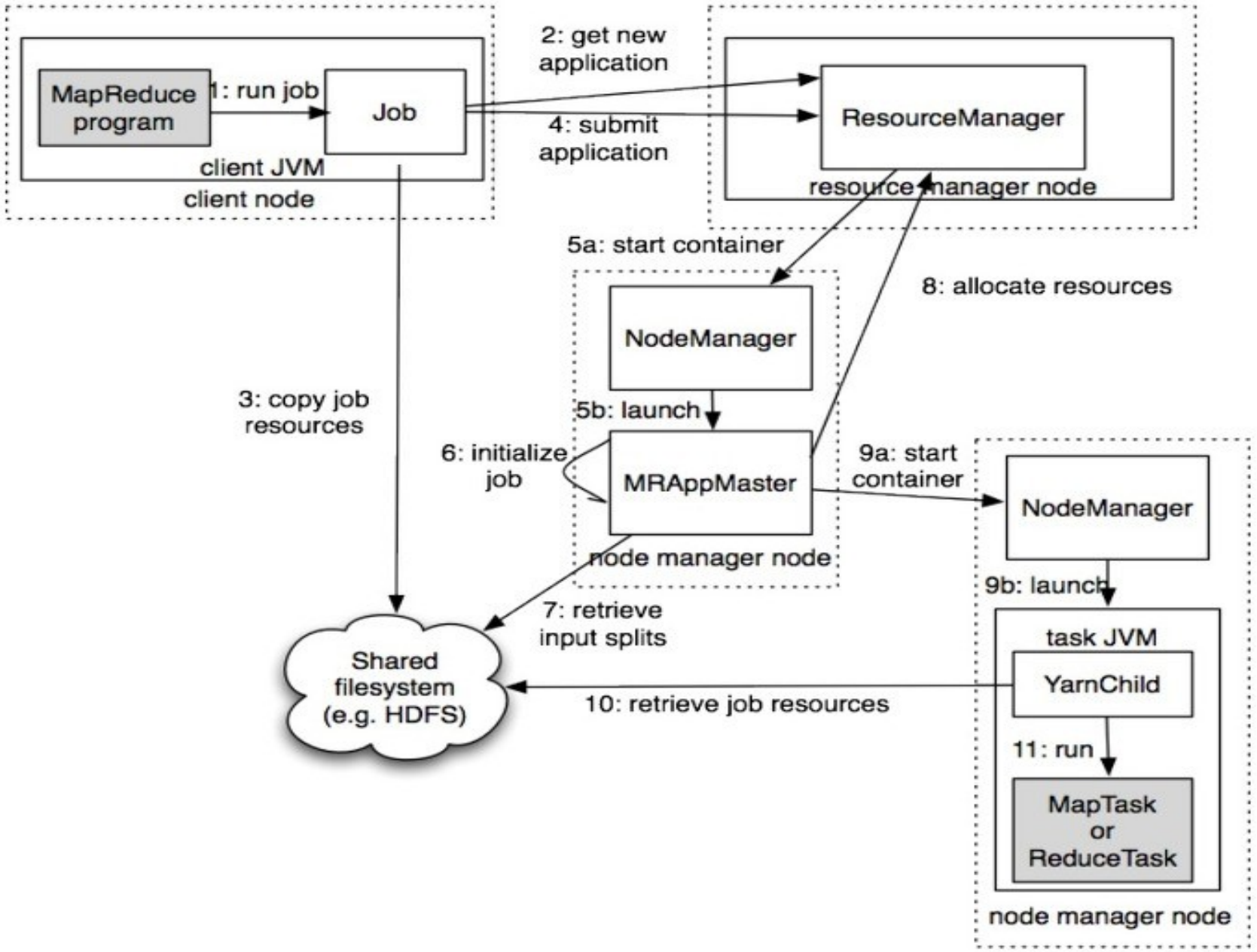
Hadoop 2. YARN

- YARN - Yet Another Resource Negotiator
- Основная идея YARN :
  - Yarn выделяет контейнеры для выполнения задач на кластере hadoop 2
  - Yarn ничего не знает про Map Reduce
  - Yarn не отслеживает ход выполнения задач
- Архитектура YARN
  - Узлы бывают двух видов :
    - ResourceManager — отслеживает запущенные контейнеры и выделяет новые.
    - Node manager — запускает контейнеры приложений
  - Приложение може запросить контейнер у Resouce Manager и запустить в нем любую задачу

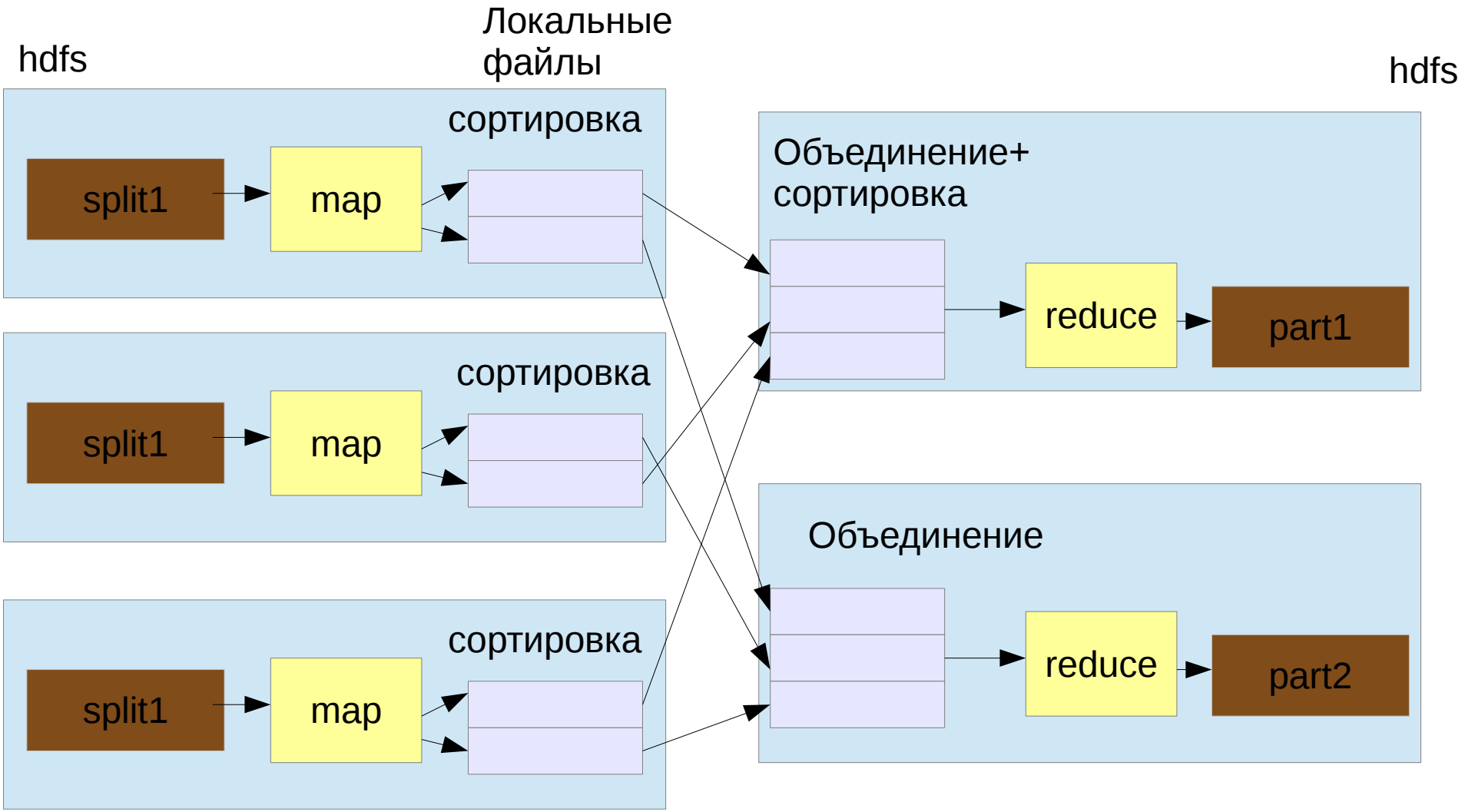
Архитектура YARN



Map Reduce в YARN



Hadoop map reduce



InputFormat

- Основные функции :
  - Разбивает исходные данные на splits
  - Обеспечивает чтение данных из split в процессе работы mapper
- Базовые классы :

```
public abstract class InputFormat<K, V> {
    public abstract List<InputSplit> getSplits(JobContext context)..;
    public abstract RecordReader<K, V> createRecordReader(
        InputSplit split,
        TaskAttemptContext context) ..
}
public abstract class InputSplit {
    public abstract long getLength() throws IOException, InterruptedException;
    public abstract String[] getLocations() throws ...;
}
```

API.Сериализация.

- Сериализация — преобразование объекта в байтовый поток данных
- Десериализация — преобразование потока байтов в объект.
- Используется как для RPC вызовов так и для сохранения промежуточных результатов в файл.
- Базовый интерфейс объекта сериализации в Hadoop — org.apache.hadoop.io.Writable

```
public interface Writable {
  void write(DataOutput out) throws IOException;
  void readFields(DataInput in) throws IOException;
}
```

- Для основных типов данных есть классы :
- IntWritable, StringWritable и т.д.

Api.Mapper

- Базовый класс для реализации mapper : org.apache.hadoop.mapreduce.Mapper

  класс параметризуется типами : <KEYIN,VALUEIN,KEYOUT,VALUEOUT>
- Базовый метод для перегрузки

  protected void map(KEYIN key, VALUEIN value, Mapper.Context context)
- Для того чтобы добавить результат функции map требуется вызвать

  context.write(KEYOUT key, VALUEOUT value)

Пример Mapper

- public class WordMapper extends Mapper<LongWritable, Text, Text, IntWritable> {

  @Override

  protected void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException {

  String line = value.toString();

  String[] words = StringTools.split(StringTools.removeAllNonSymbols(line).toLowerCase());

  for (String word : words) {

  context.write(new Text(word), new IntWritable(1));

  }

  }

  }

Api. Reducer

- Базовый класс для реализации mapper : org.apache.hadoop.mapreduce.Reducer

  класс параметризуется типами : <KEYIN,VALUEIN,KEYOUT,VALUEOUT>
- Базовый метод для перегрузки

  protected void reduce(KEYIN key, Iterable<VALUEIN> values, Reducer.Context context)
- Для того чтобы добавить результат функции map требуется вызвать

  context.write(KEYOUT key, VALUEOUT value)

Пример Reducer

```
public class WordReducer extends Reducer<Text, IntWritable, Text, LongWritable> {
    @Override
   protected void reduce(Text key, Iterable<IntWritable> values, Context context) throws
IOException, InterruptedException {
        long count=0;
        Iterator iter = values.iterator();
        while(iter.hasNext()) {
            iter.next();
            count++;
        }
        context.write(key, new LongWritable(count));
    }
}
```

Запуск приложения

```
public class WordCountApp {
    public static void main(String[] args) throws Exception {
        if (args.length != 2) {
            System.err.println("Usage: WordCountApp <input path> <output path>");
            System.exit(-1);
        }
        Job job = Job.getInstance();
        job.setJarByClass(WordCountApp.class);
        job.setJobName("Word count");
        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));
        job.setMapperClass(WordMapper.class);
        job.setReducerClass(WordReducer.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);
       System.exit(job.waitForCompletion(true) ? 0 : 1);
    }
}
```

Отказоустойчивость

- Отказ Task
  - TaskTracker обнаруживает, что task перестает присылать информацию о ходе выполнения либо java процесс завершился с ошибкой.
  - Task tracker помечает task как невыполненный и сообщает об этом jobracker. JobTracker ставит задачу в очередь.
- Отказ TaskTracker
  - Все задачи помечаются в JobTracker как killed и запускаются на других TaskTracker.
- Отказ JobTracker
  - Полный перезапуск среды и задач администратором.