

# Streaming OODT:

---

Combining Apache Spark's Power with Apache OODT

---

Michael Starch – NASA Jet Propulsion Laboratory

---

# Agenda

---

- Data and Processing
- Data Systems
- Apache OODT
- Apache Spark
- Streaming OODT
- Examples
- Where can I get the code?
- Acknowledgements
- Questions

# **Data and Processing**

---

# Data and Processing

---

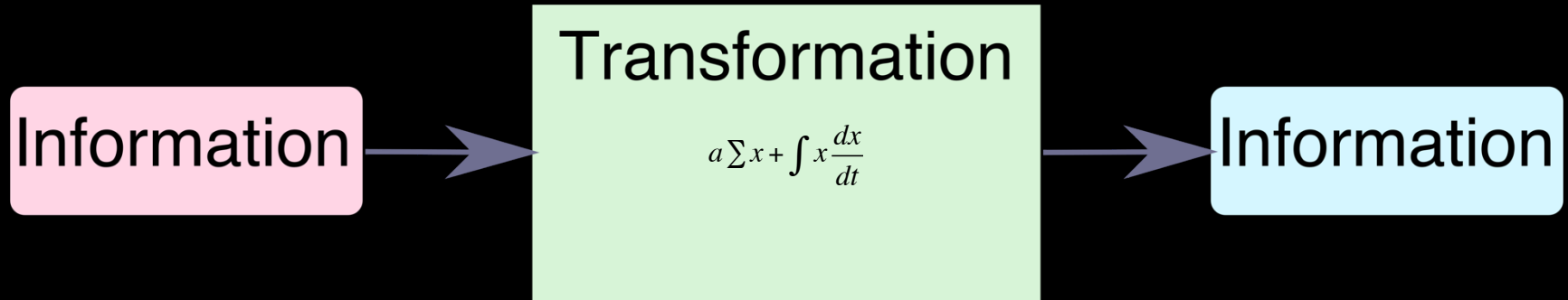


Figure 1: What is data processing?

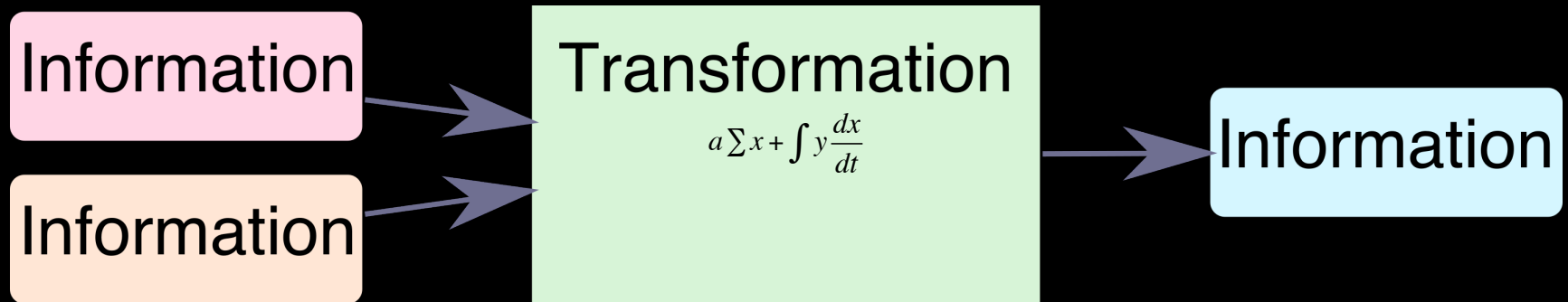


Figure 2: More complex data processing

# Parallelization

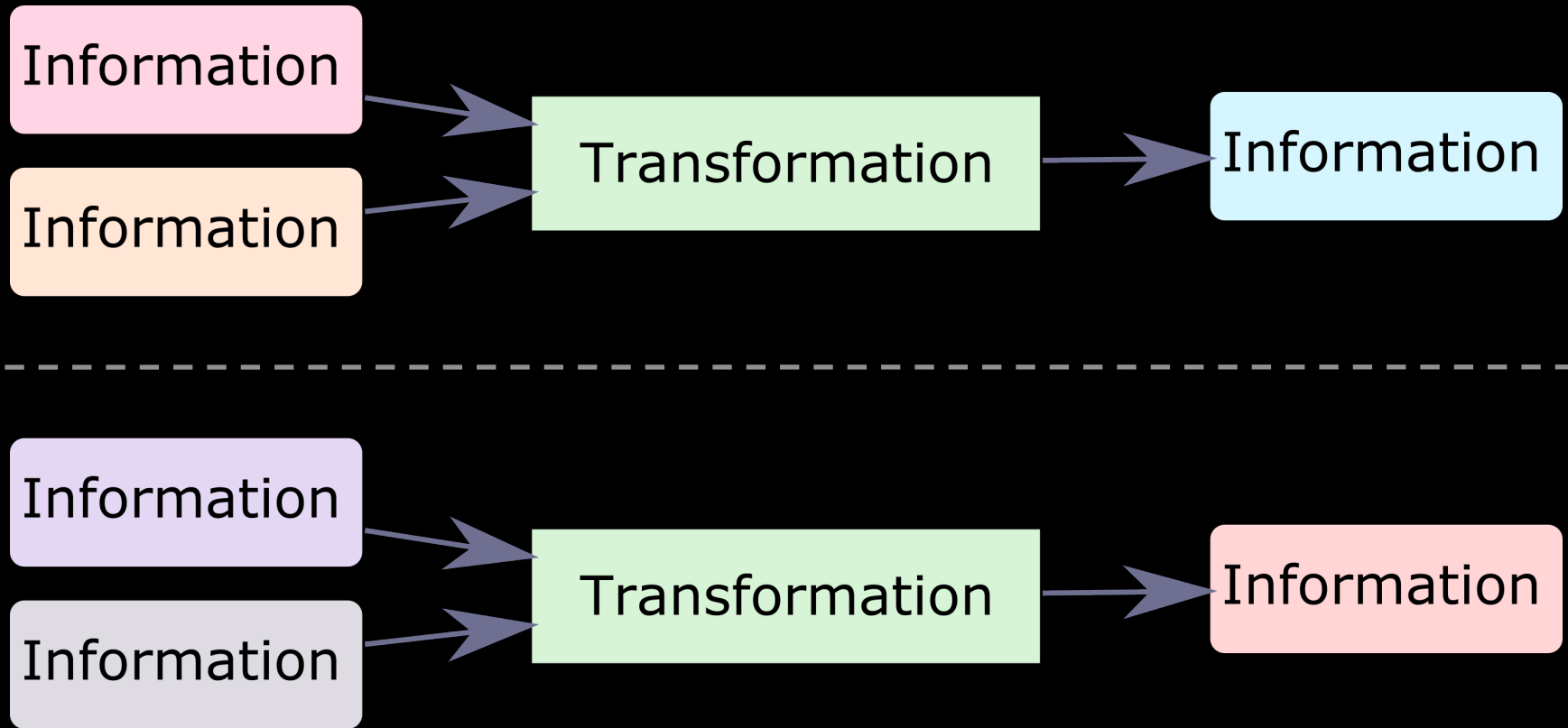


Figure 3: Parallelizing data processing

# Big Data

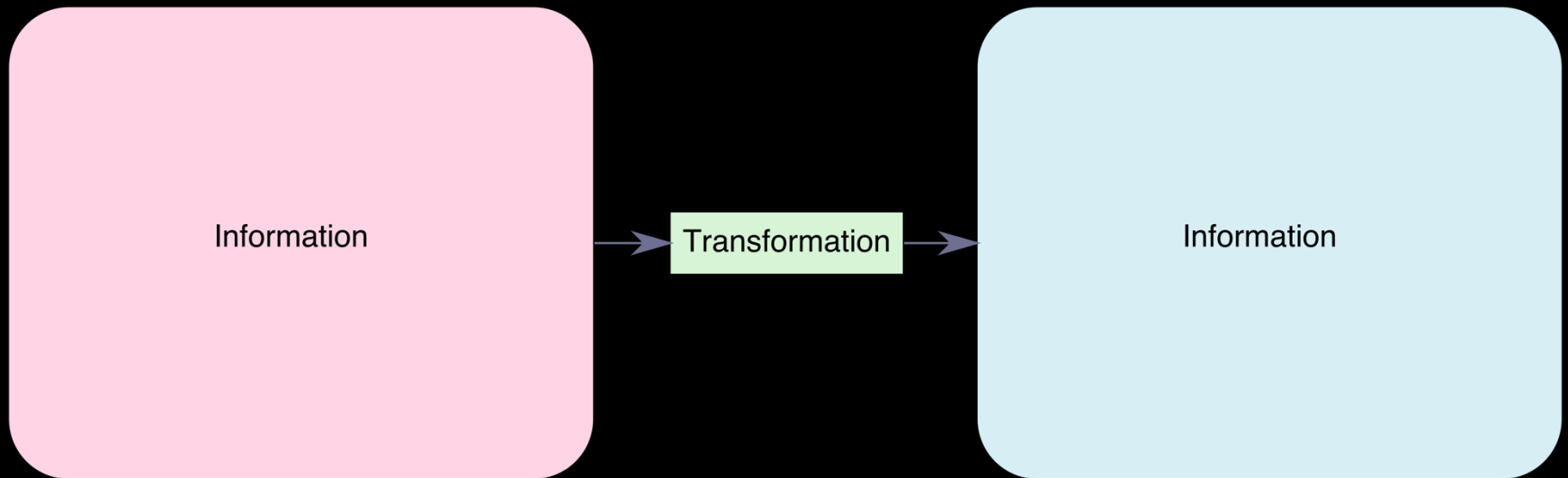


Figure 4: Data is becoming very large

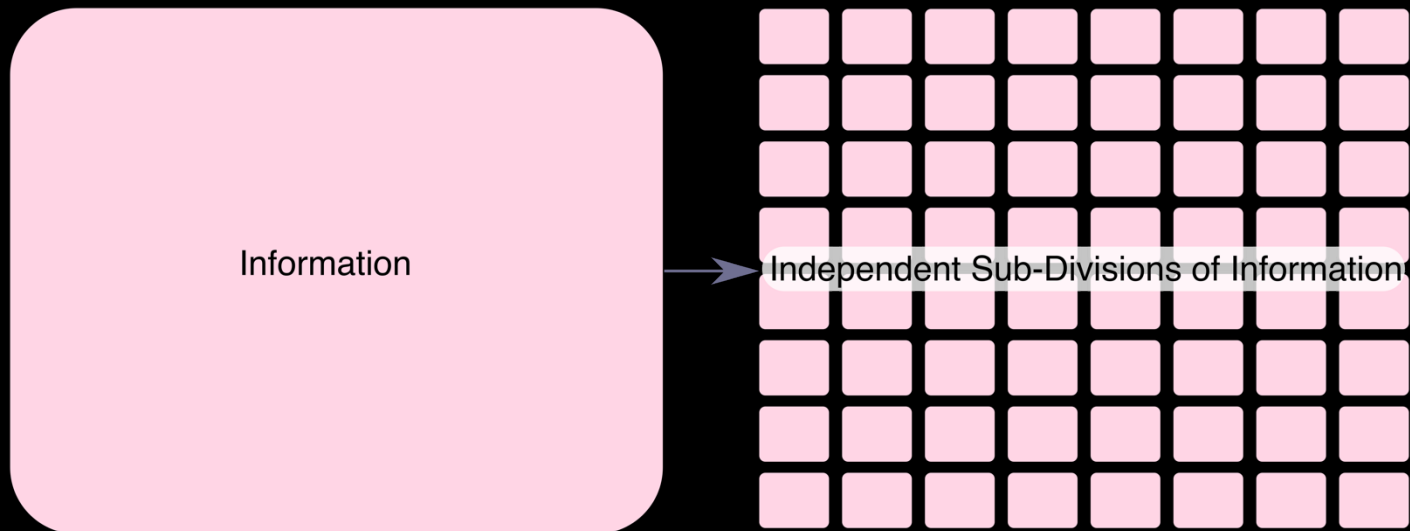


Figure 5: Parallelizable big-data

# **Data Systems**

---

# Archival and Search

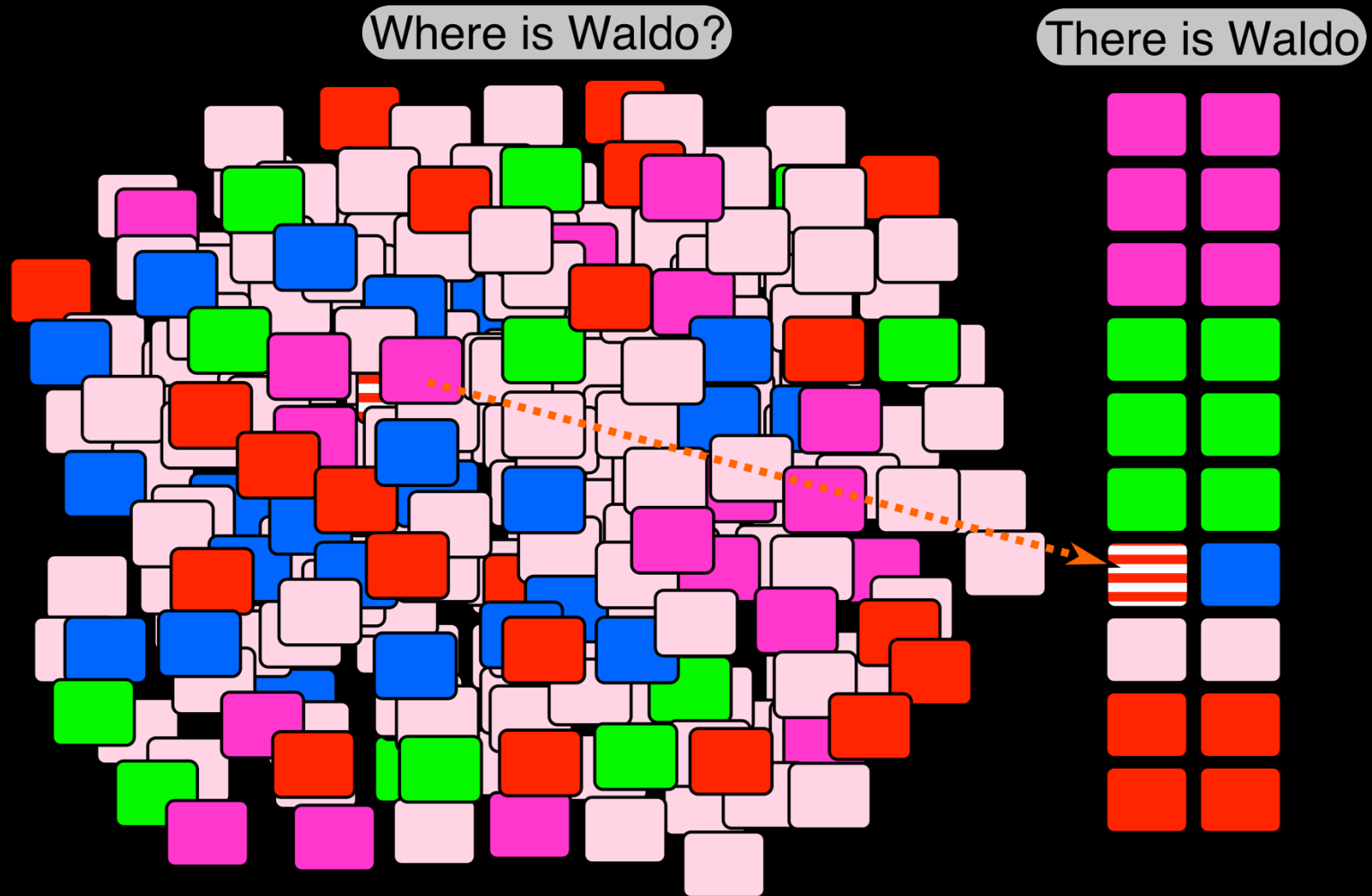


Figure 6: Archiving and searching in data sets

# Processing and Resource Management

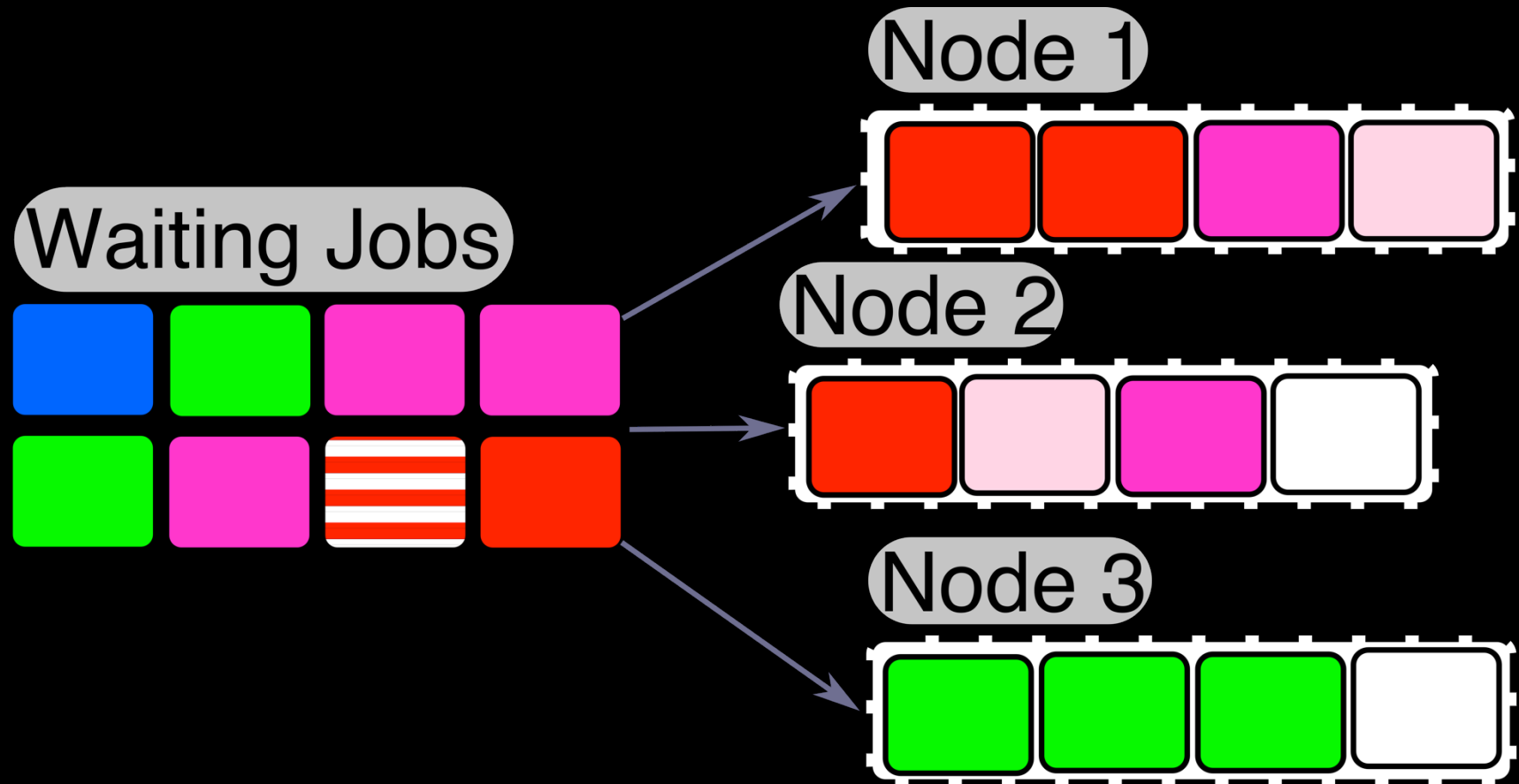


Figure 7: Processing and resource management

## Data Ingest and Delivery

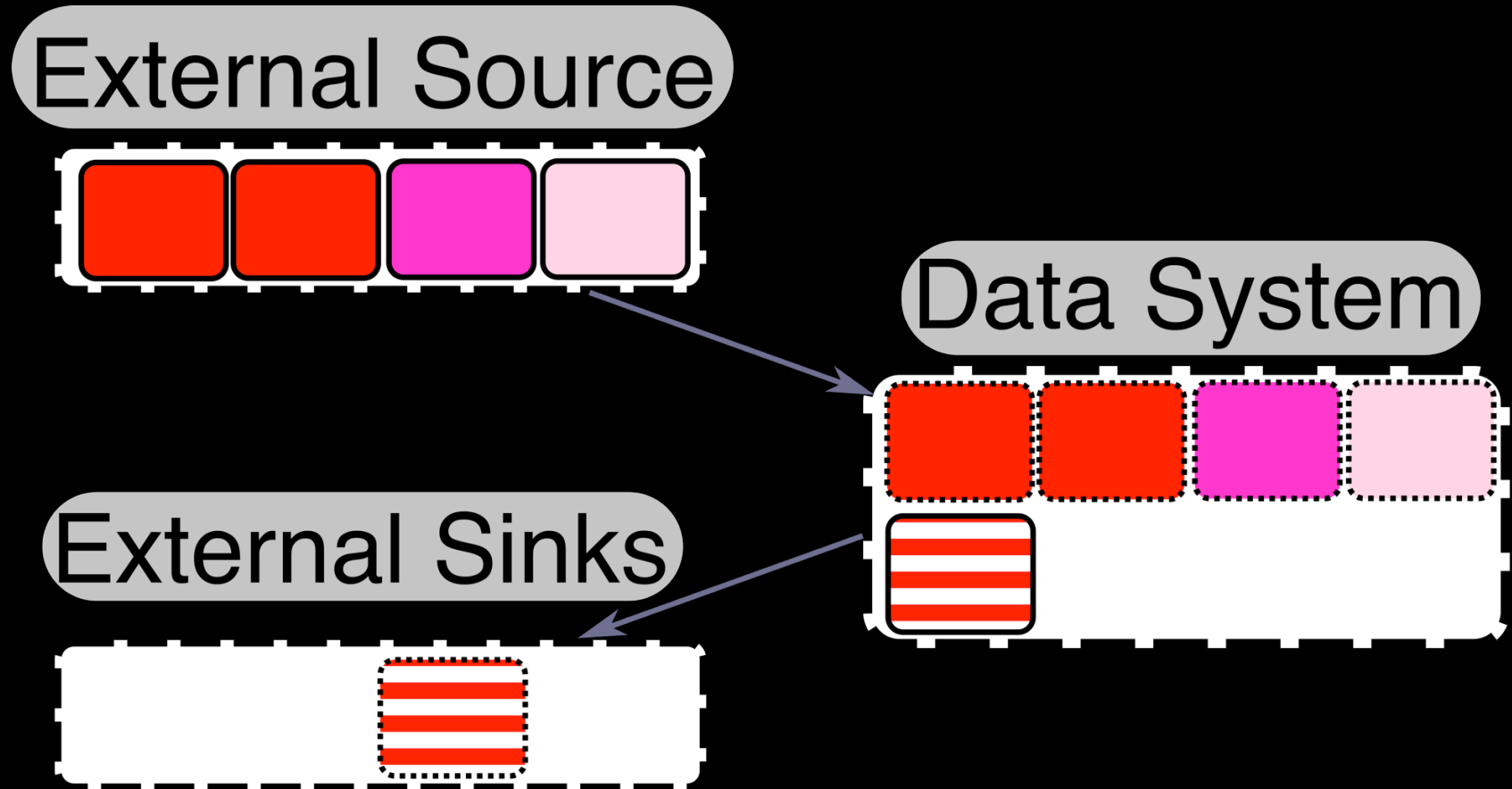


Figure 8: Data ingestion and delivery

# **Apache OODT**

---

# Apache OODT

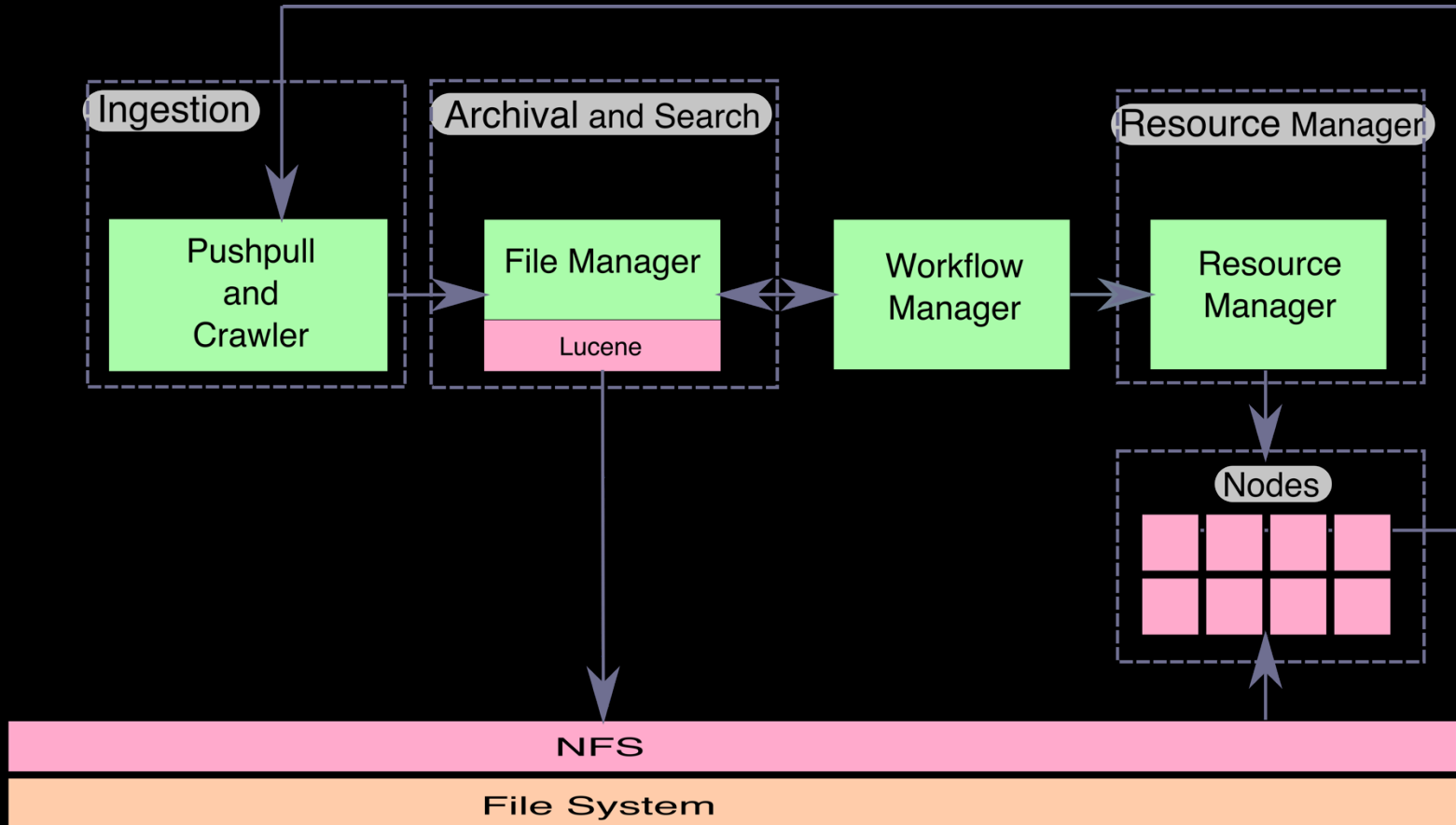


Figure 9: Base Object-Oriented Data Technology (OODT)

# Archival and Search

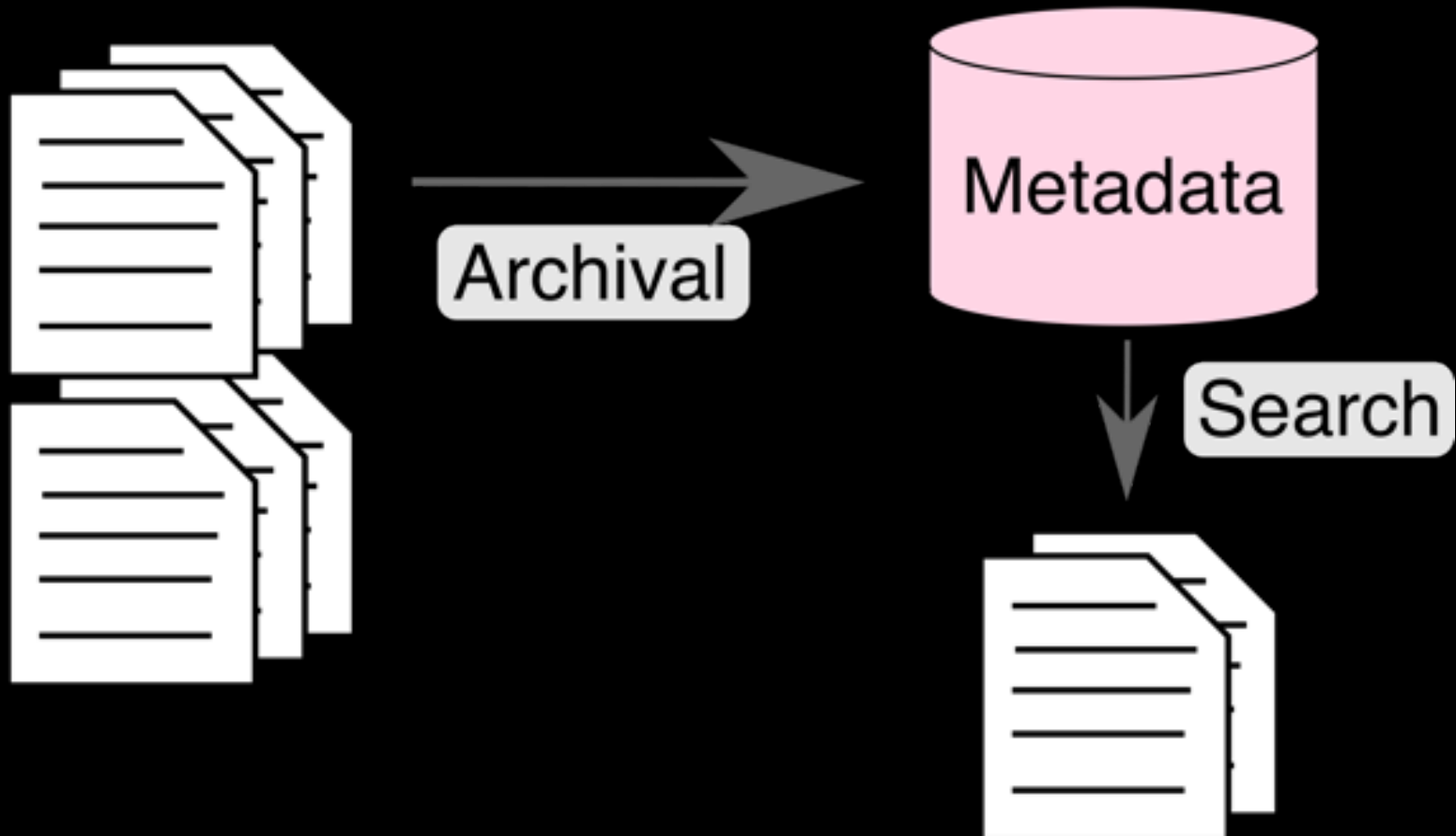


Figure 10: OODT metadata-based search

# Workflow Management

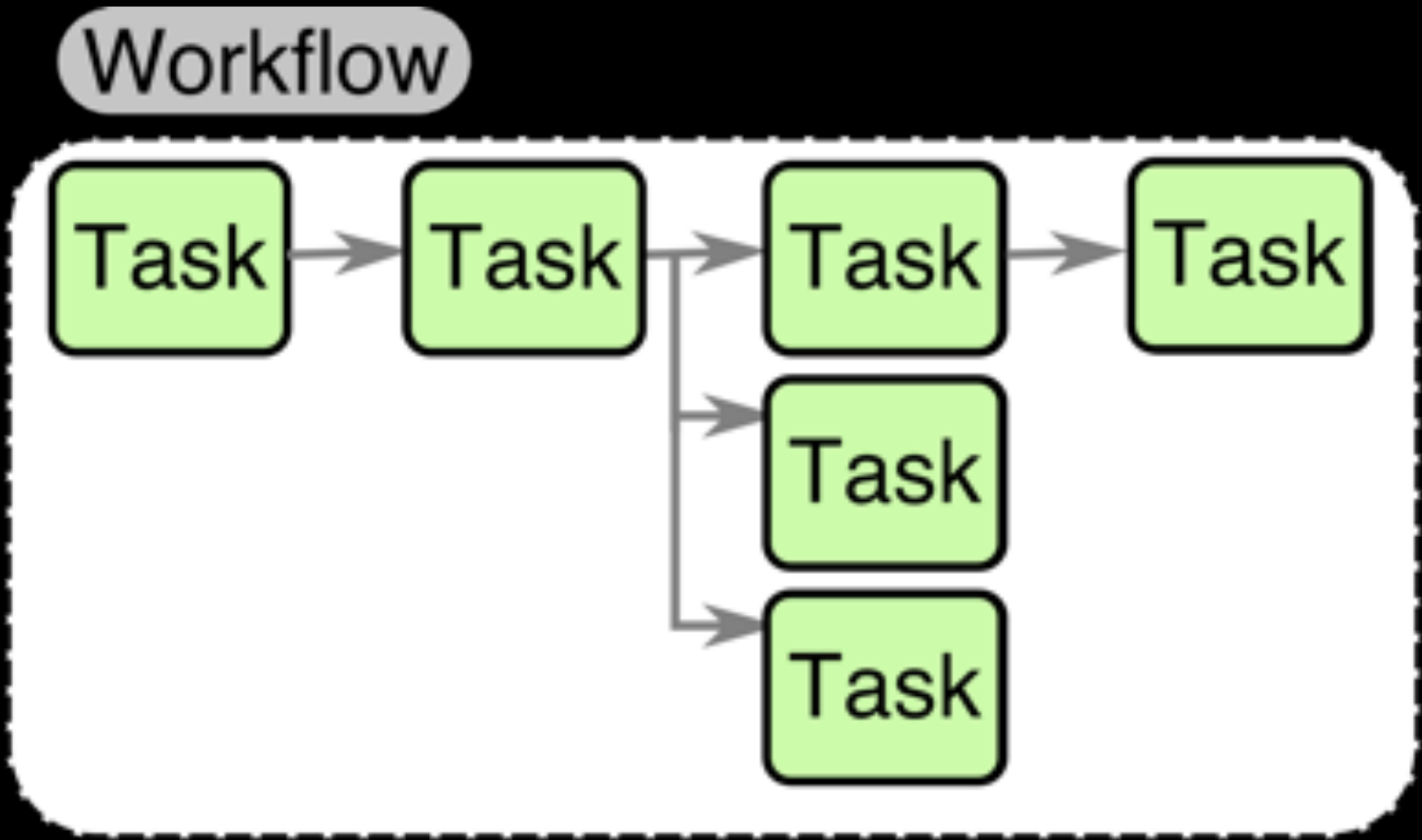


Figure 11: OODT workflow management

# Limitations

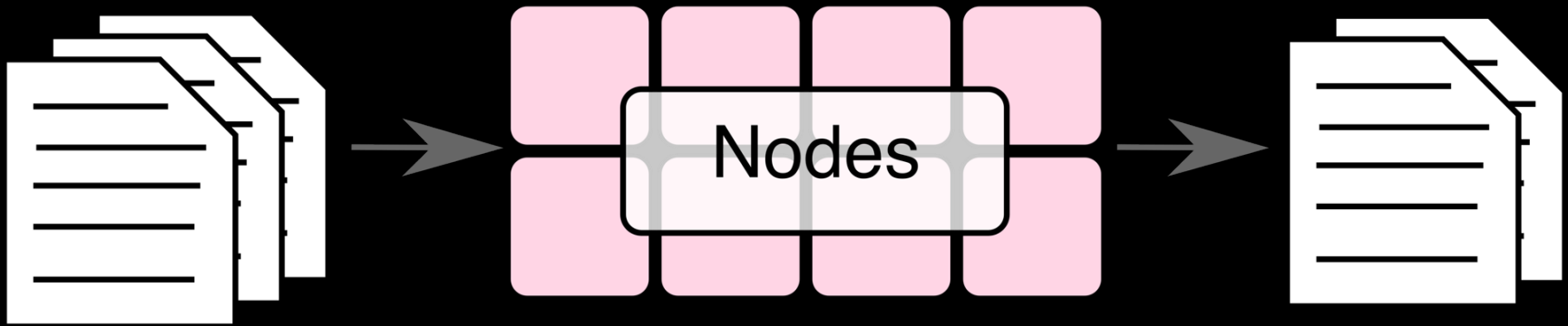


Figure 12: Simplified OODT Architecture

# Apache Spark

---

# Map Reduce Processing

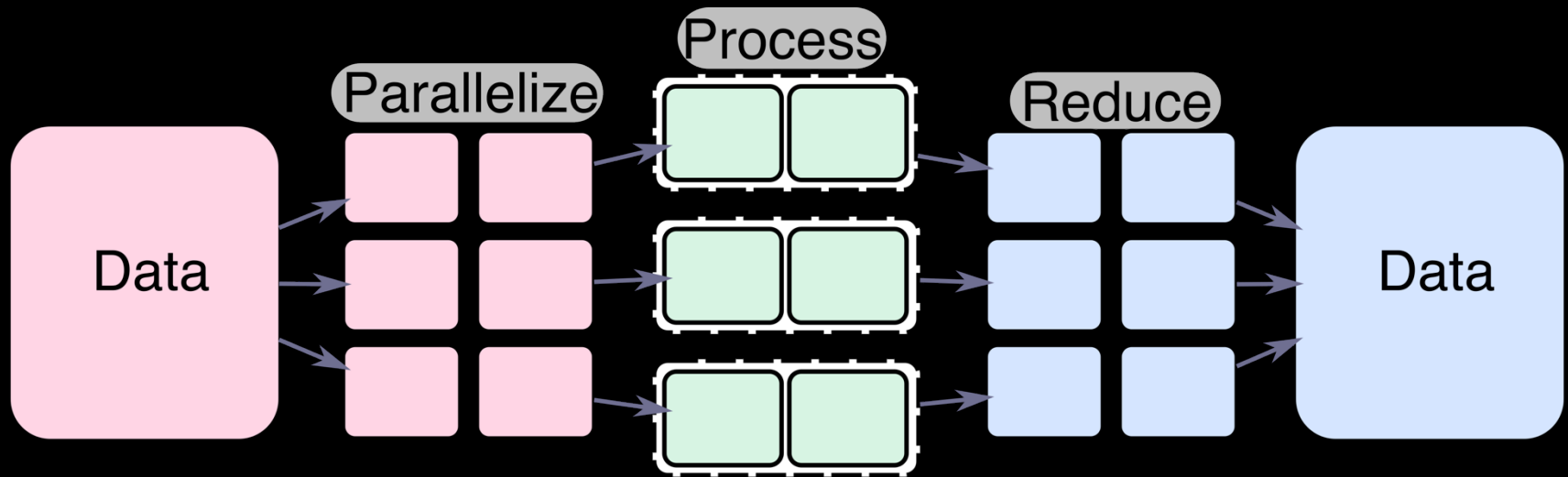


Figure 13: Map Reduce Processing

# Berkley Data Analysis Stack

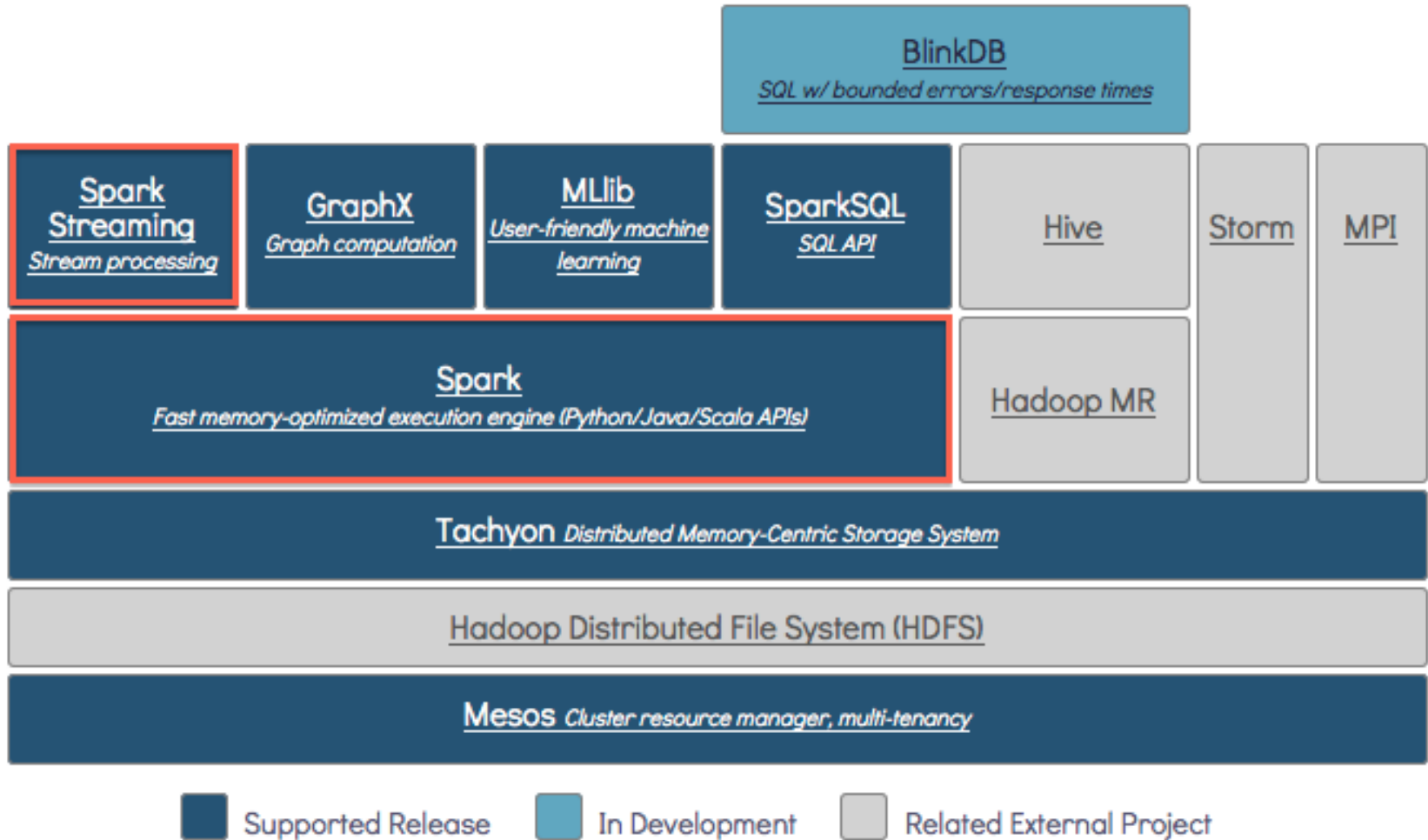


Figure 14: Berkley data analysis stack components Source: <https://amplab.cs.berkeley.edu/software/>

# Apache Spark

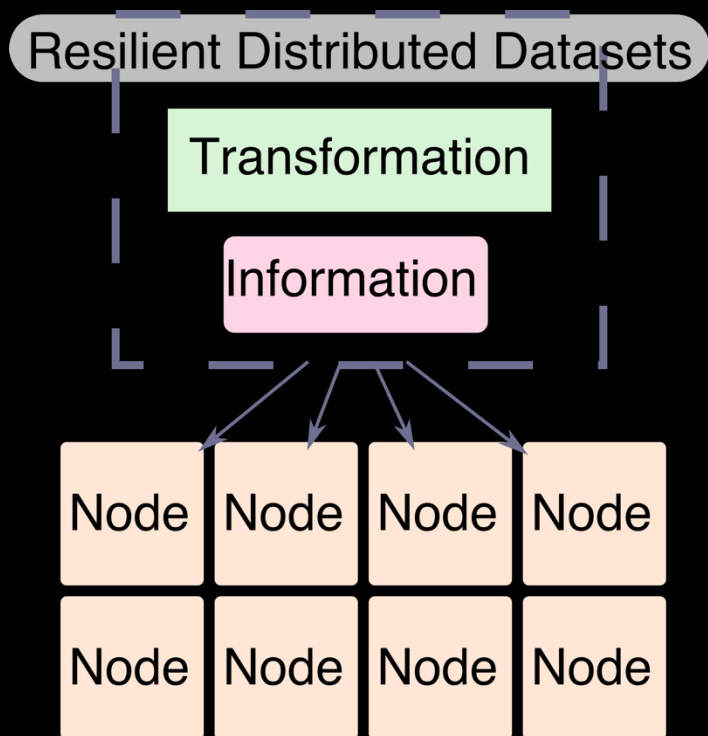


Figure 15: Resilient Distributed Datasets

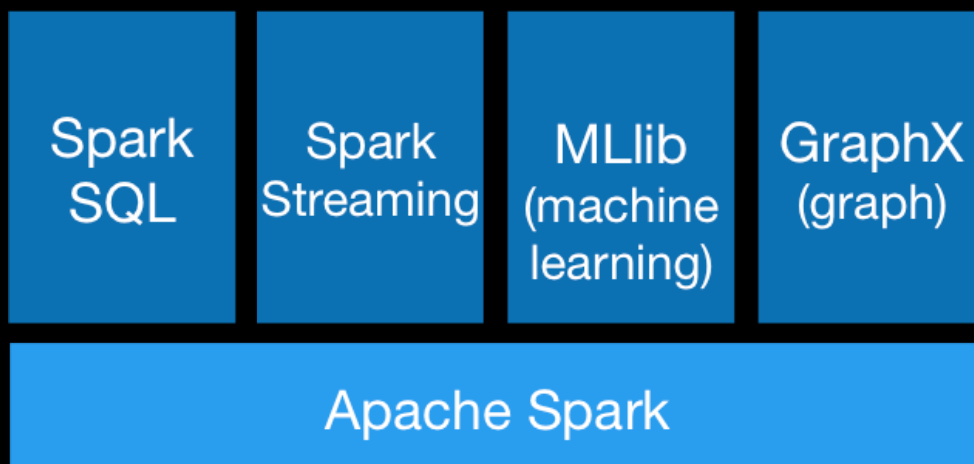


Figure 16: Apache Spark libraries

Source: <https://spark.apache.org/images/spark-stack.png>

# Streaming OODT

---

# Streaming OODT Design

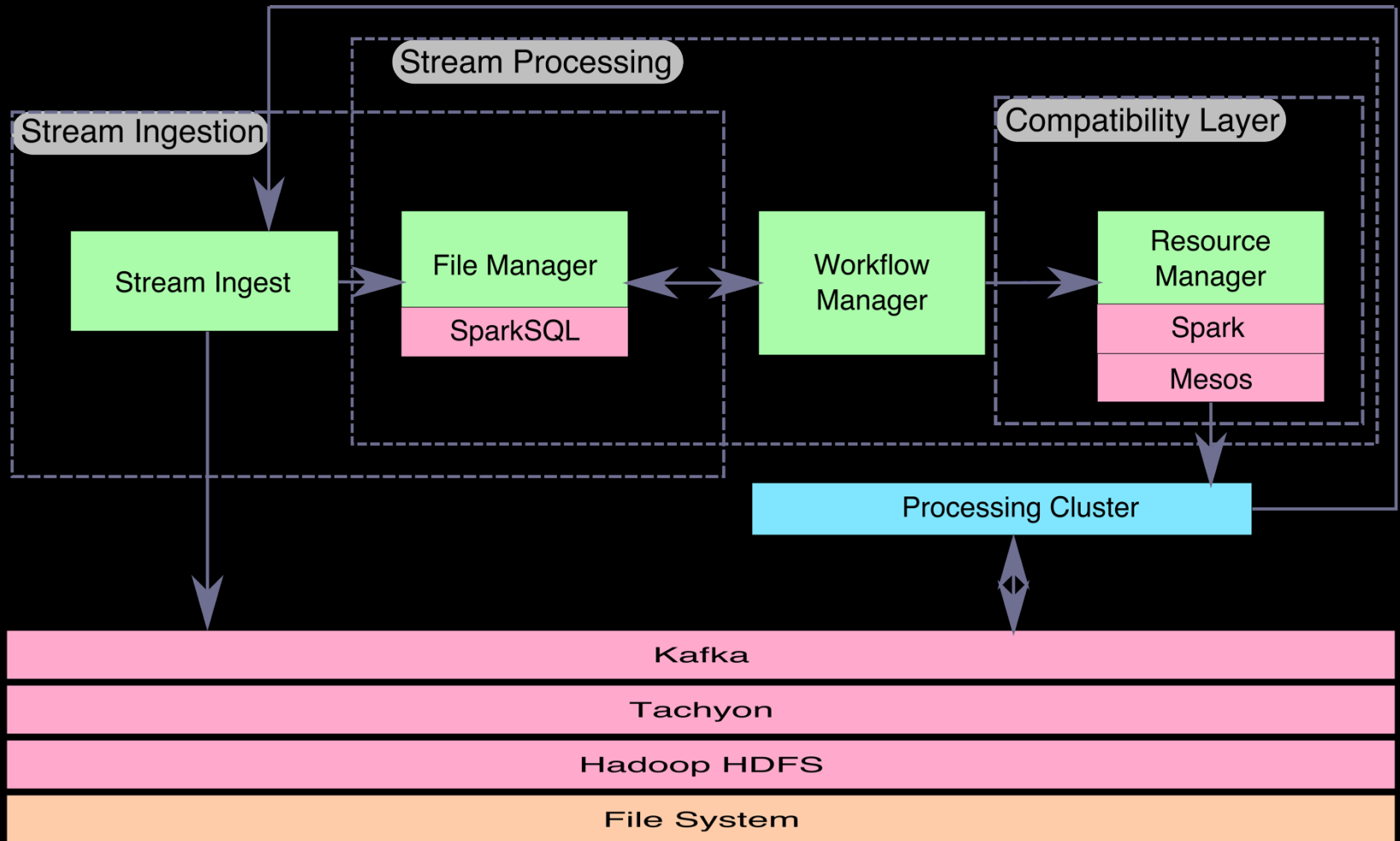


Figure 17: Design and implementation of Streaming OODT

# Modified Architecture

---

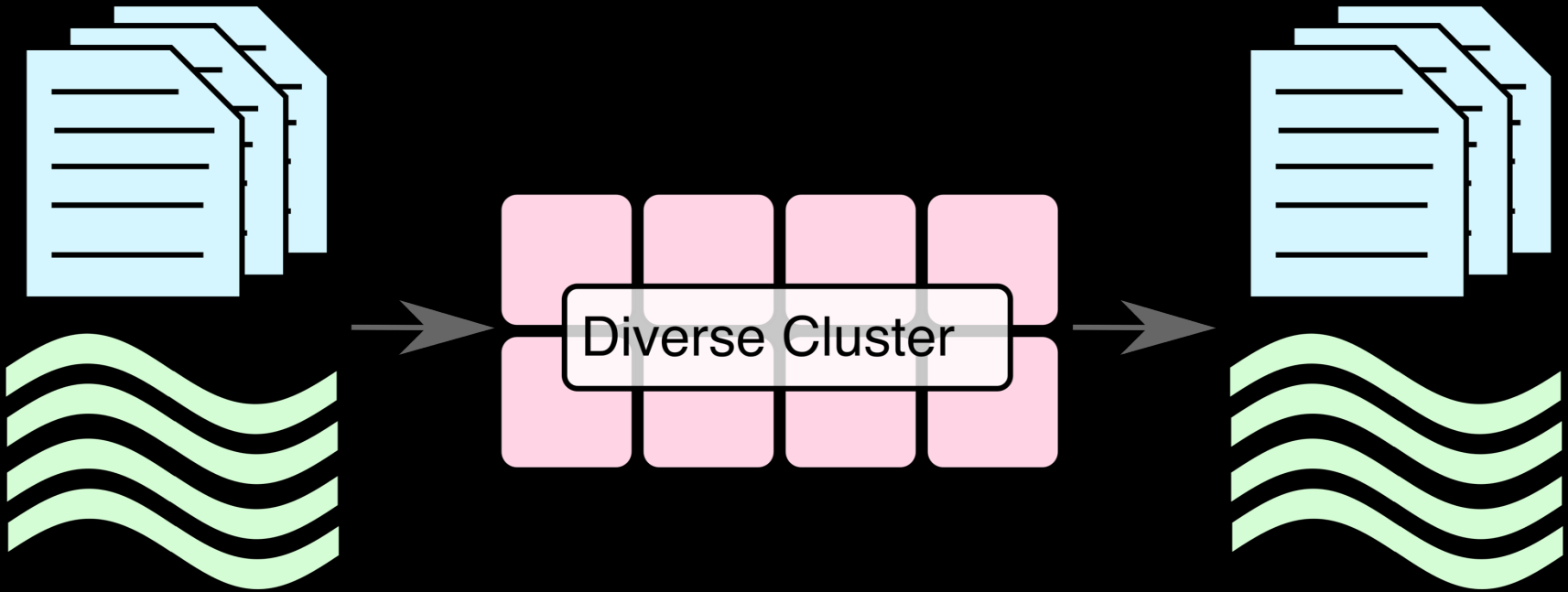


Figure 18: Improved OODT Architecture for big-data processing

# Examples

---

# Example - Palindromes

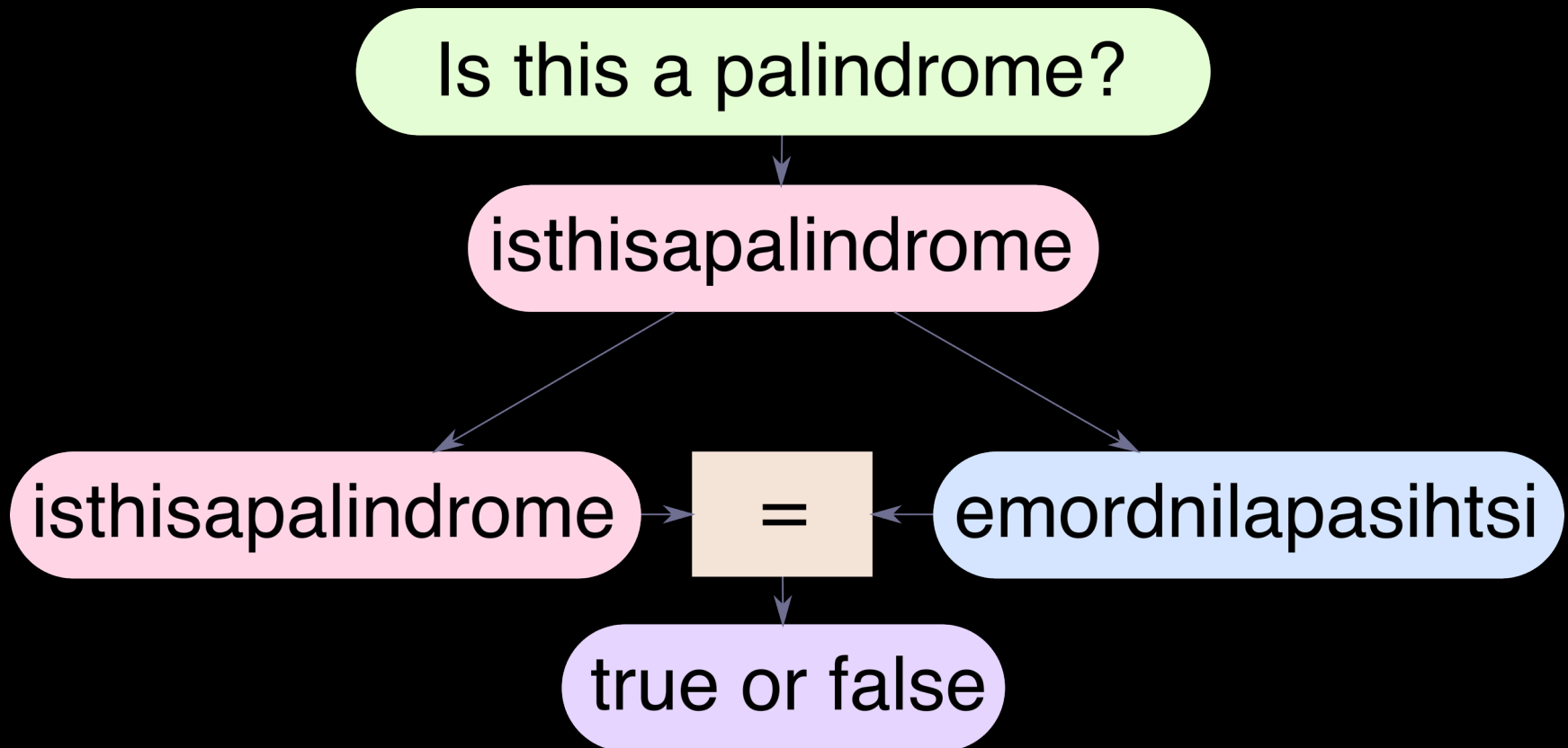


Figure 19: Palindrome detection algorithm

# Example - Code

---

```
//Example detection algorithm
...
public static boolean isPalindrome(String line) {
    line = line.replaceAll("\\s", "").toLowerCase();
    return line.equals(new StringBuilder(line).reverse().toString());
}
...
//Spark wrapper class for detection algorithm
static class FilterPalindrome implements Function<String, Boolean>
{
    public Boolean call(String s) {
        return isPalindrome(s);
    }
}
...
```

Sample 1: Palindrome detection shared code

# Example – Data Set

clowring infratrochanteric unlimitable overstaffing ...  
nonsubstantiality incongeniality ghbor  
gargil semiconventionality betokens clinodome ...  
pulviniform actualize cousins moocha Mosaism craals  
midstout desightment Boehmenism LP ravelins underskirt CSB  
cossas xen- nonlucidness unvagrantness togata  
noncaptiousness dromioid lambie undergarments salvages...  
LAP  
revealableness outsnore headstalls metallography  
outgazed unstintingly boongary provinces trans-Mongolian...

Sample 2: Palindrome file sample

...

10,805,887,353 Bytes (11 GB)

46284 palindromes

# Example – Shootout

~~Spark~~

429.774s

1 CPU

```
//Sample java code
...
String file =
    input.getValue("file");
br = new BufferedReader(new
    FileReader(file));
String line;
while ((line = br.readLine())
    != null) {
    if (PalindromeUtils
        .isPalindrome(line))
        count++;
}
...
```

Sample 3: Naïve file processing code

Spark

16.72s

~92 CPUs

```
//Sample java code
...
JavaRDD<String> rdd = sc.textFile(
    input.getValue("file"));
JavaRDD<String> filtered =
    rdd.filter(new PalindromeUtils
        .FilterPalindrome());
long count = filtered.count();
...
```

Sample 4: Spark file processing code

# Example - Streaming

```
JavaReceiverInputDStream<String> stream =
    ssc.socketTextStream(input.getValue("host"),
        Integer.parseInt(input.getValue("port")));
JavaDStream<String> filtered = stream.filter(new
    PalindromeUtils.FilterPalindrome());
final JavaDStream<Long> count = filtered.count();
/* Begin: output code */
count.foreachRDD(new Function<JavaRDD<Long>,Void>(){
    public Void call(JavaRDD<Long> jrdd) throws Exception {
        synchronized(output) {
            Long[] collected = (Long[])jrdd.rdd().collect();
            for (Long item : collected)
                output.println("Found "+item.longValue()+ " palindromes.");
        }
        return null;}});
/* End: output code*/
ssc.start();
ssc.awaitTermination();
```

# Example – Streaming Configuration

---

```
...  
<instanceClass name=  
"org.apache.oodt.cas.resource.spark.examples.StreamingPalindromeEx  
ample" />  
<inputClass name=  
  "org.apache.oodt.cas.resource.structs.NameValueJobInput">  
  <properties>  
    <property name="host" value="host" />  
    <property name="port" value="7007" />  
    <property name="time" value="60000" />  
    <property name="output" value="/home/user/files/output-  
streaming-palindrome.txt" />  
  </properties>  
</inputClass>  
<queue>quick</queue>  
<load>1</load>  
...
```

Sample 6: Streaming palindromes configuration

# Example – Streaming In Action

---

```
Found 34 palindromes.  
Found 118 palindromes.  
Found 115 palindromes.  
█
```

```
Found 34 palindromes.  
Found 118 palindromes.  
Found 115 palindromes.  
Found 90 palindromes.  
Found 103 palindromes.  
█
```

```
Found 34 palindromes.  
Found 118 palindromes.  
Found 115 palindromes.  
Found 115 palindromes.  
Found 90 palindromes.  
Found 103 palindromes.  
Found 124 palindromes.  
Found 117 palindromes.  
Stopping after 60 seconds.  
█
```

# Where can I get the code?

It's Open Source! Jump on in!

Apache OODT SVN:

<https://svn.apache.org/repos/asf/oodt/trunk/>

Mailing List:

[dev@oodt.apache.org](mailto:dev@oodt.apache.org)

# Acknowledgments

---

## NASA Jet Propulsion Laboratory

---

Research & Technology Development

“Archiving, Processing and Dissemination for the Big Data Era”

## Apache Software Foundation

---

Apache OODT Project

Avez-vous des questions?

你  
有  
沒  
有  
問  
題  
？

Haben Sie Fragen?

Questions?

¿Tienen preguntas?