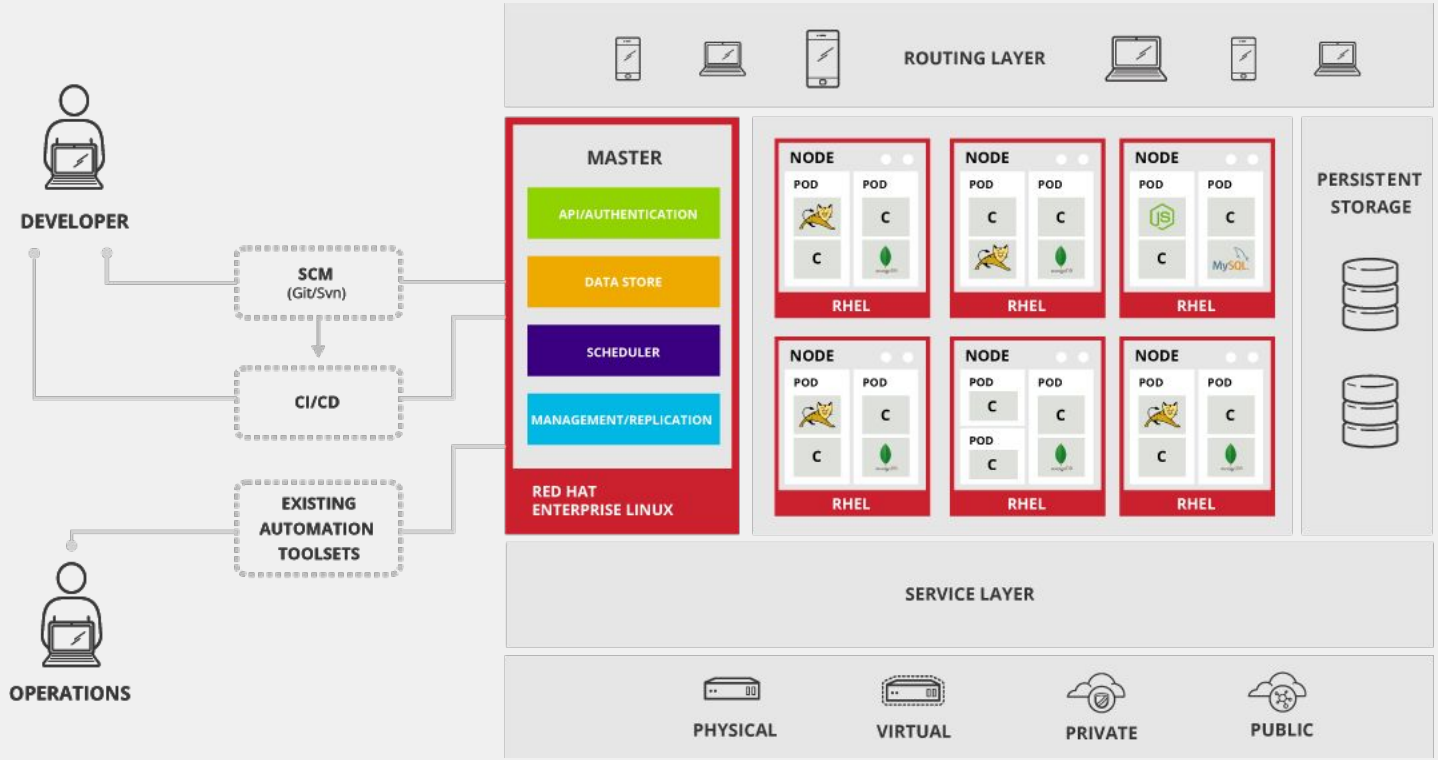
Intelligent Applications on OpenShift

from Prototype to Production

Rebecca Simmonds
Senior Software Engineer

Michael McCune
Principal Software Engineer

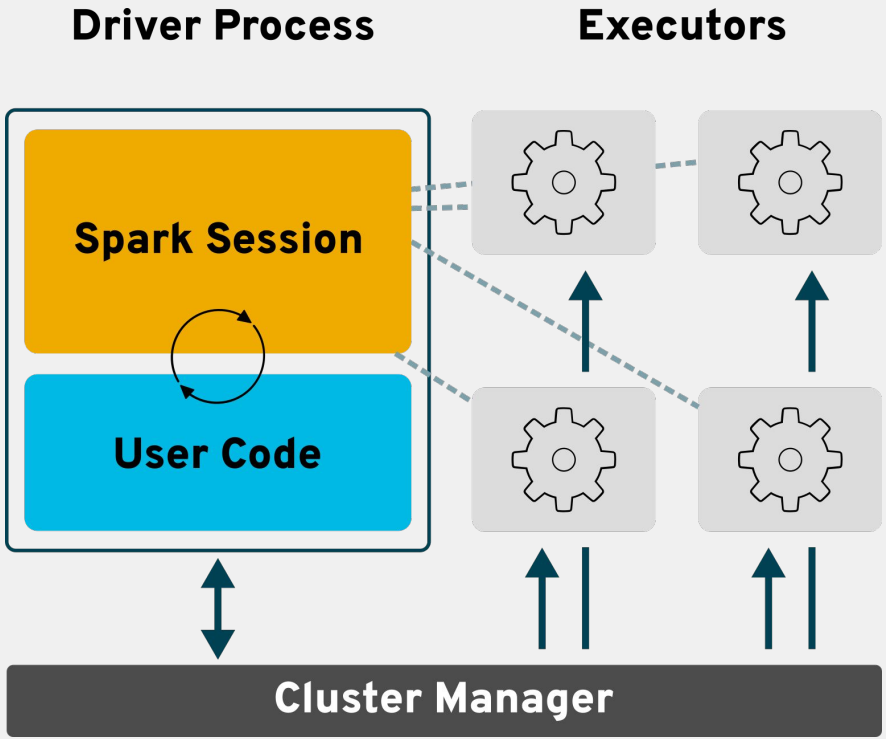
OpenShift, our foundation



Why OpenShift?

- container based orchestration
- source-to-image workflows
- application portability

Apache Spark, our engine



Why Spark?

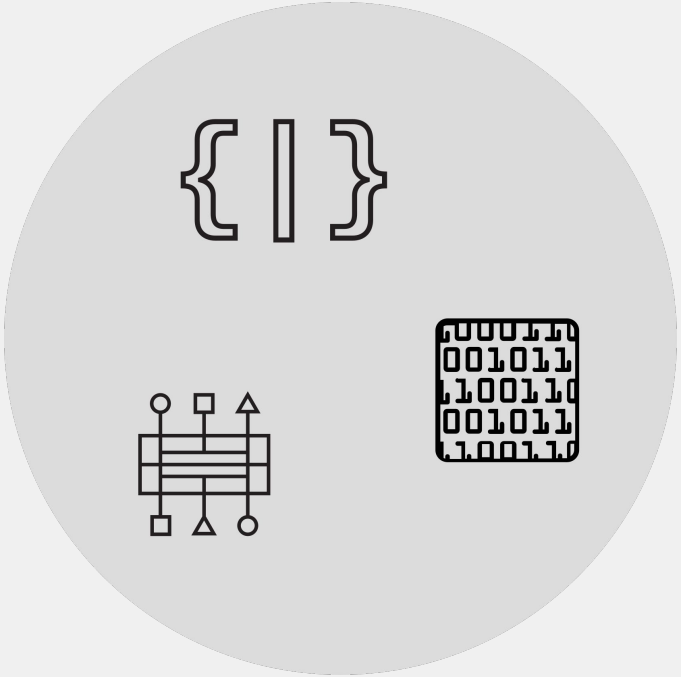
- immutable and resilient by design
- broad processing paradigm support
- rich ecosystem

Spark's fundamental abstraction

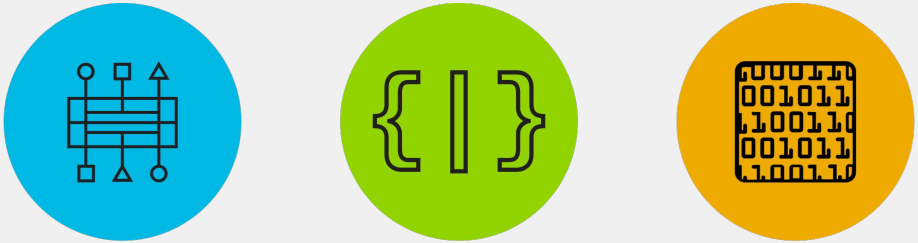
Resilient Distributed Dataset (RDD)

partitioned, lazy, and immutable homogenous collections

Microservice architectures

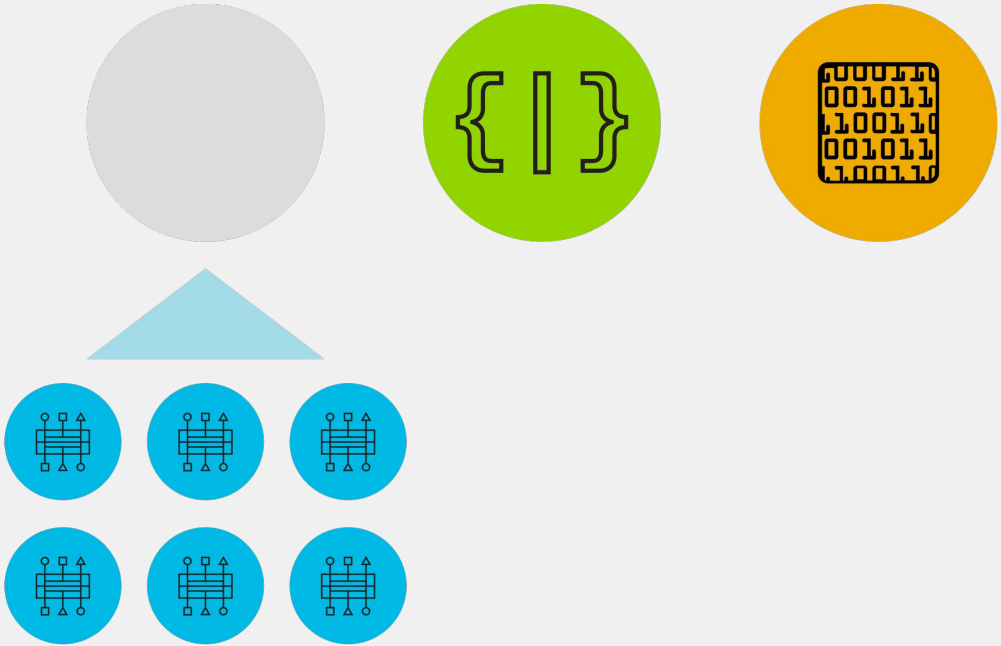


Microservice architectures

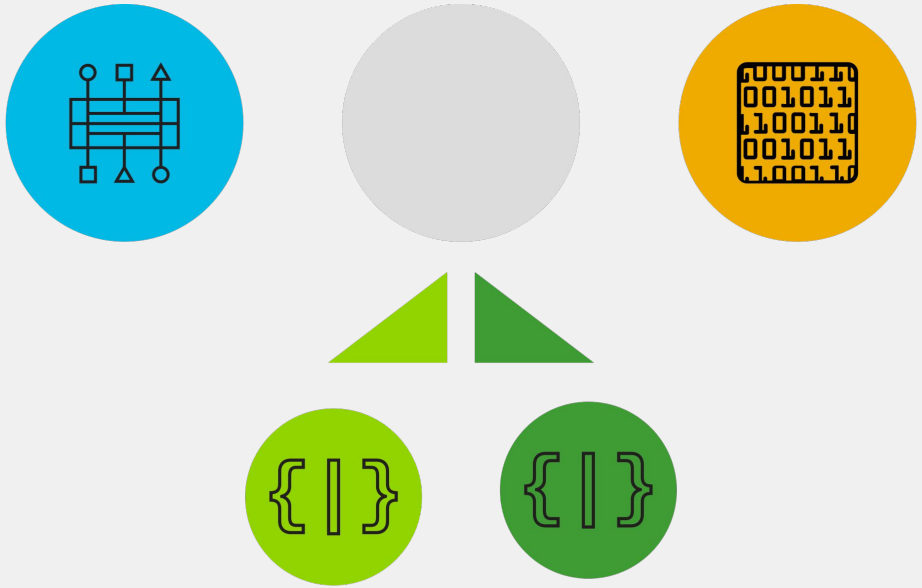


- Unix philosophy
- modular and flexible
- stateful versus stateless
- network resilient

Microservice architectures

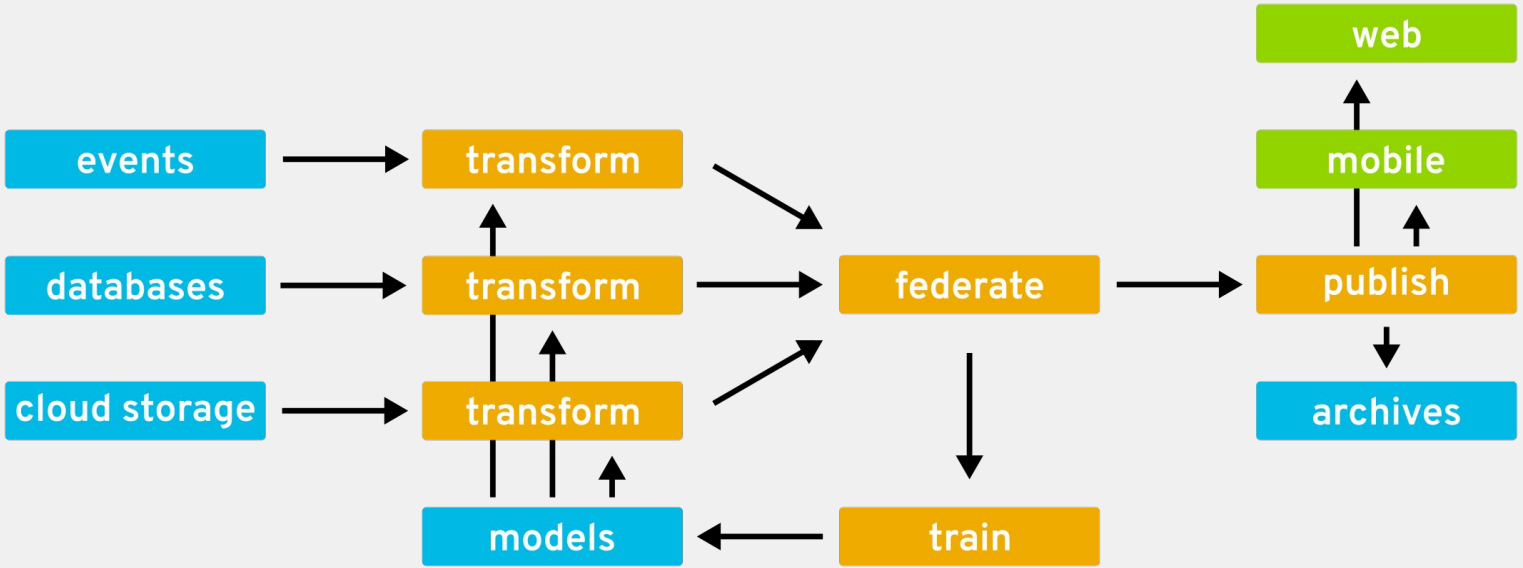


Microservice architectures

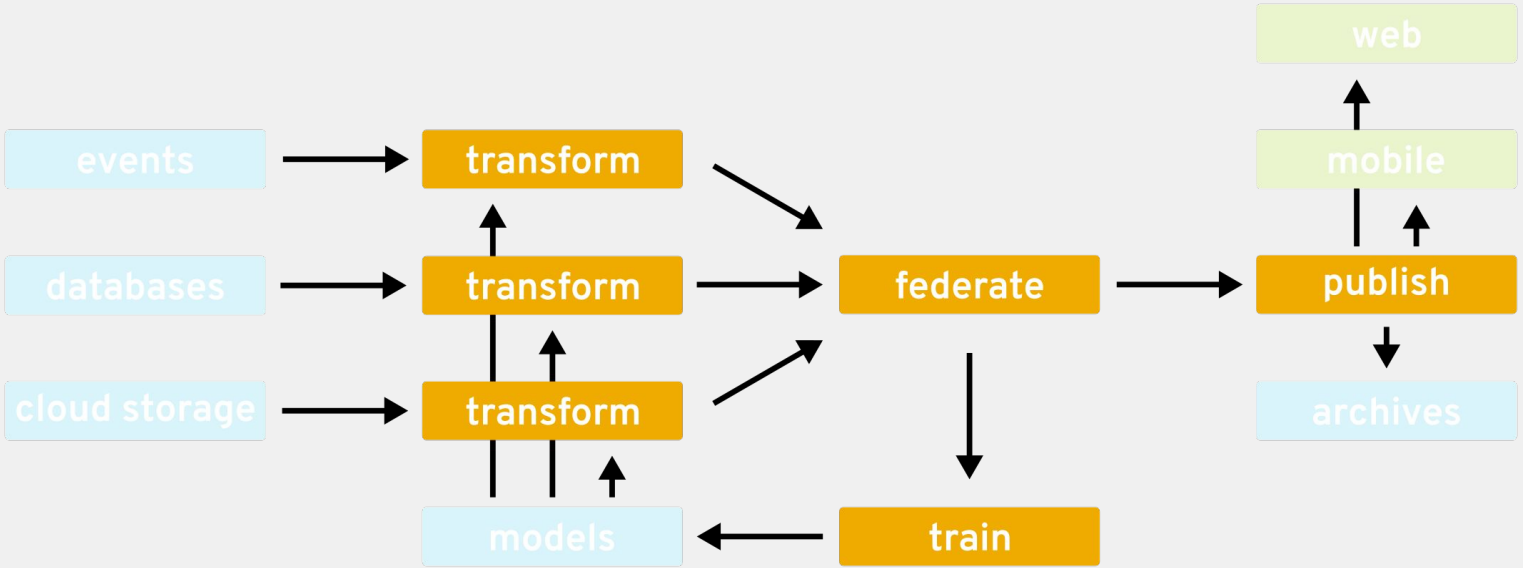


Microservice architectures

Intelligent applications



Intelligent applications



Intelligent services

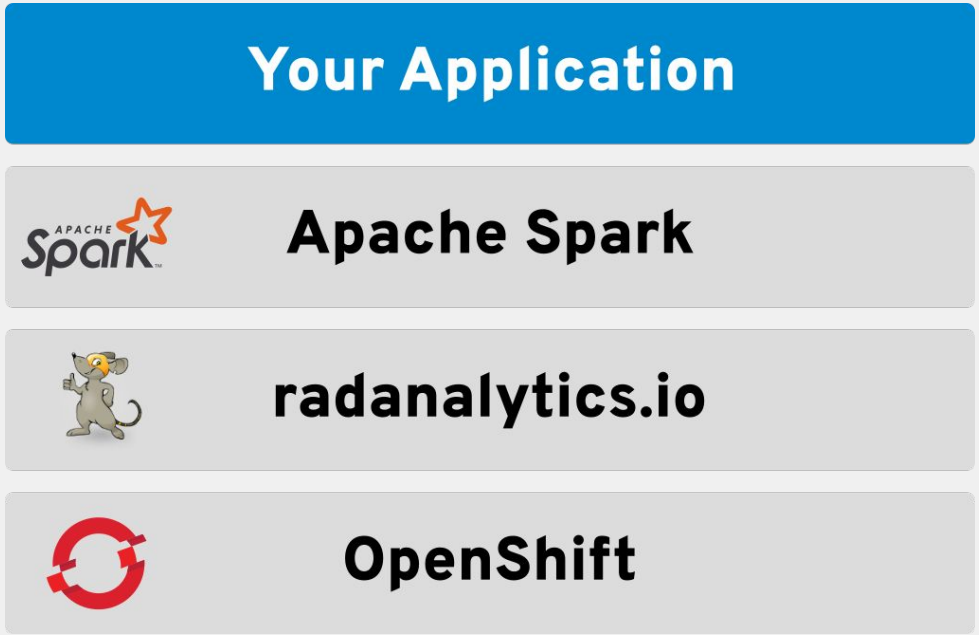
Introducing radanalytics.io

What is radanalytics.io?

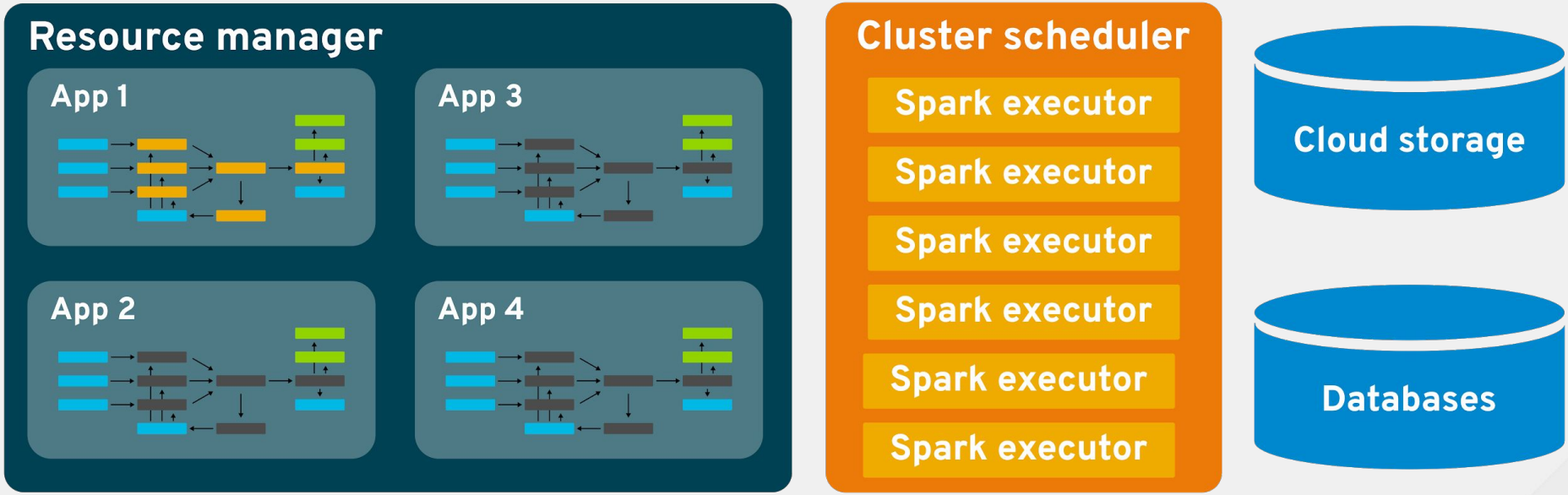
An open source community working to empower intelligent application lifecycles on OpenShift

A collection of projects to enable analytics and machine learning frameworks on OpenShift

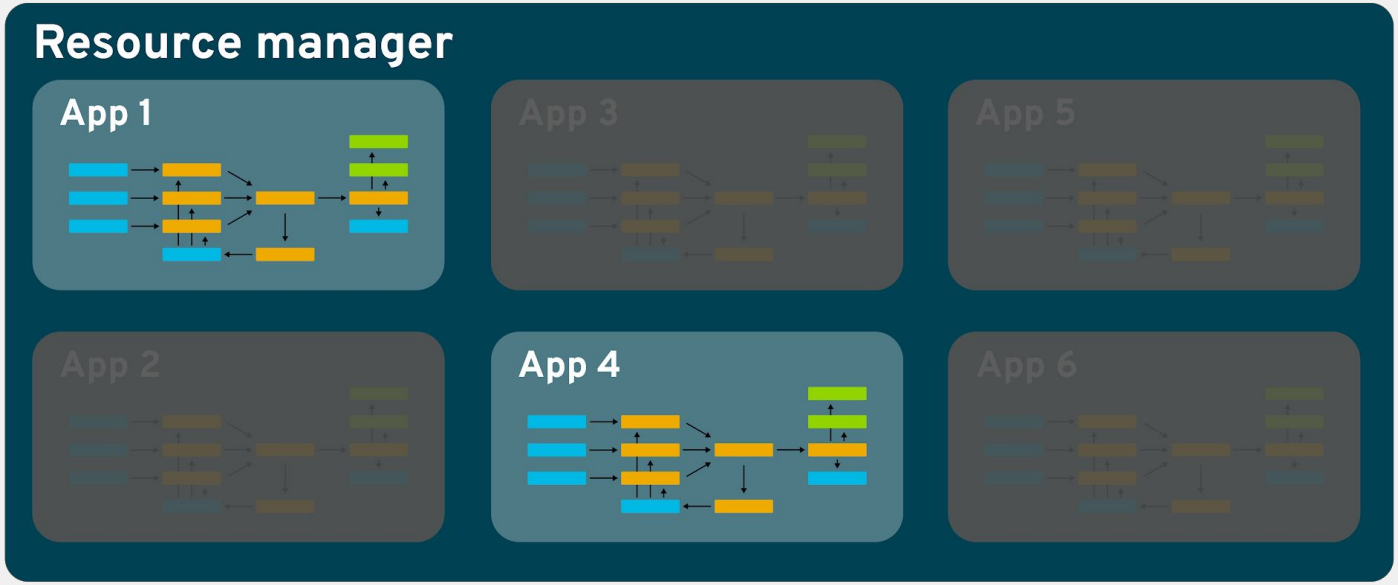
Project Oshinko



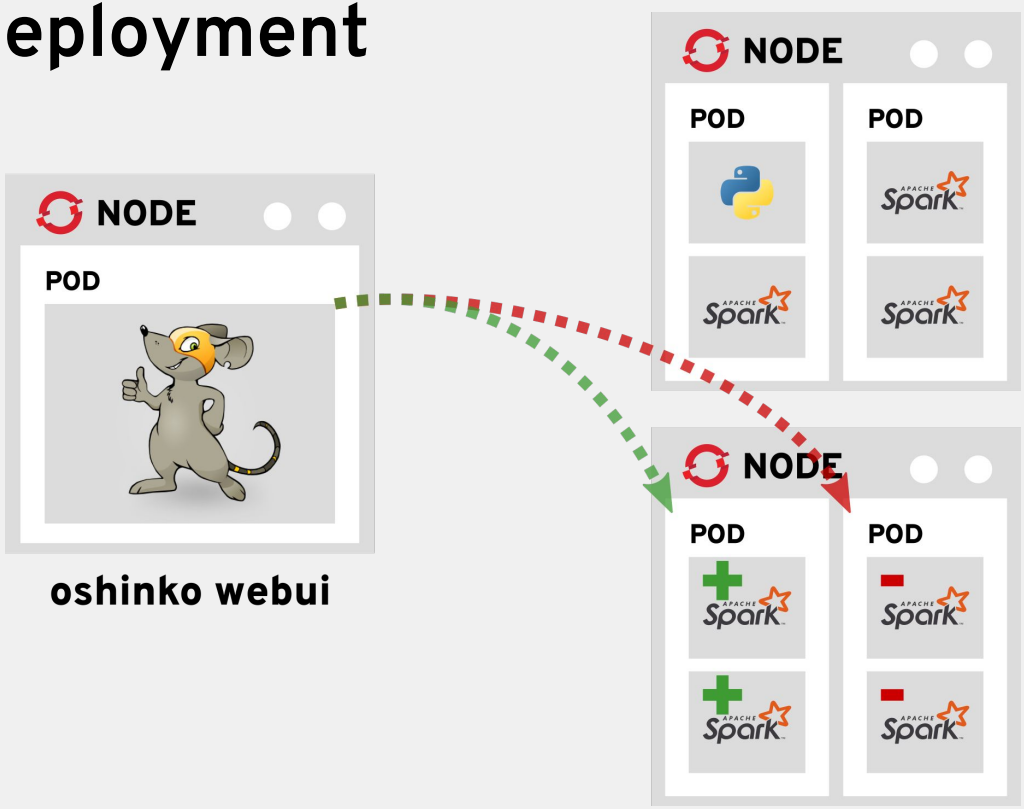
Oshinko motivation



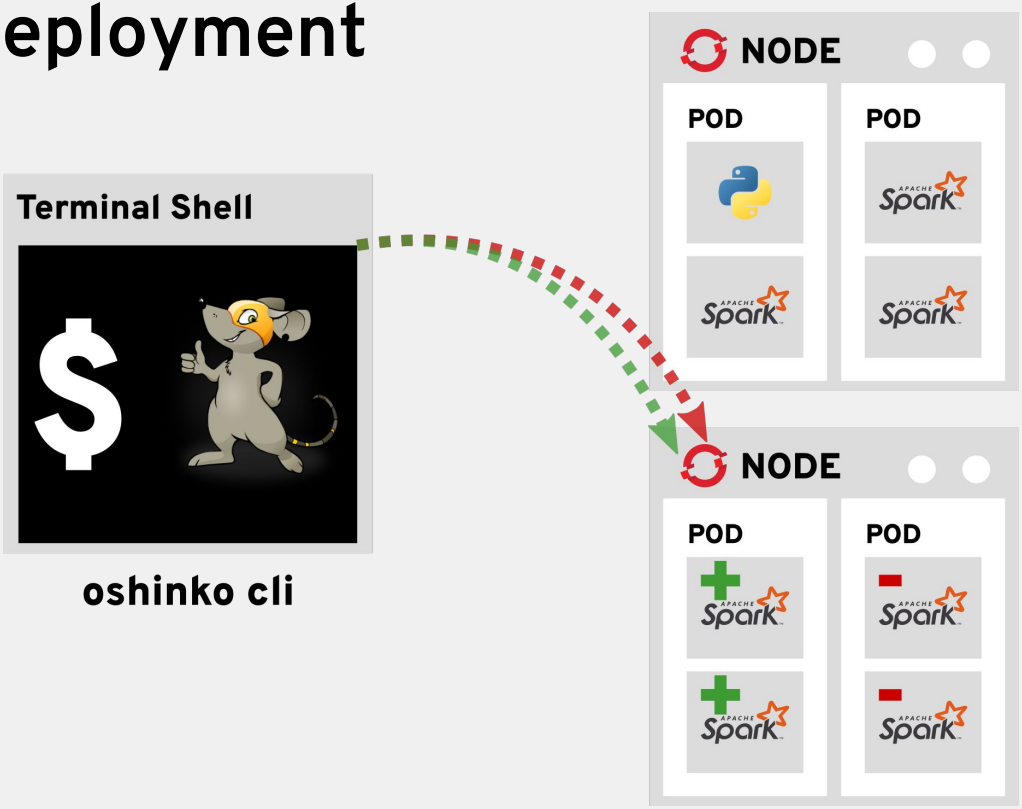
Oshinko motivation



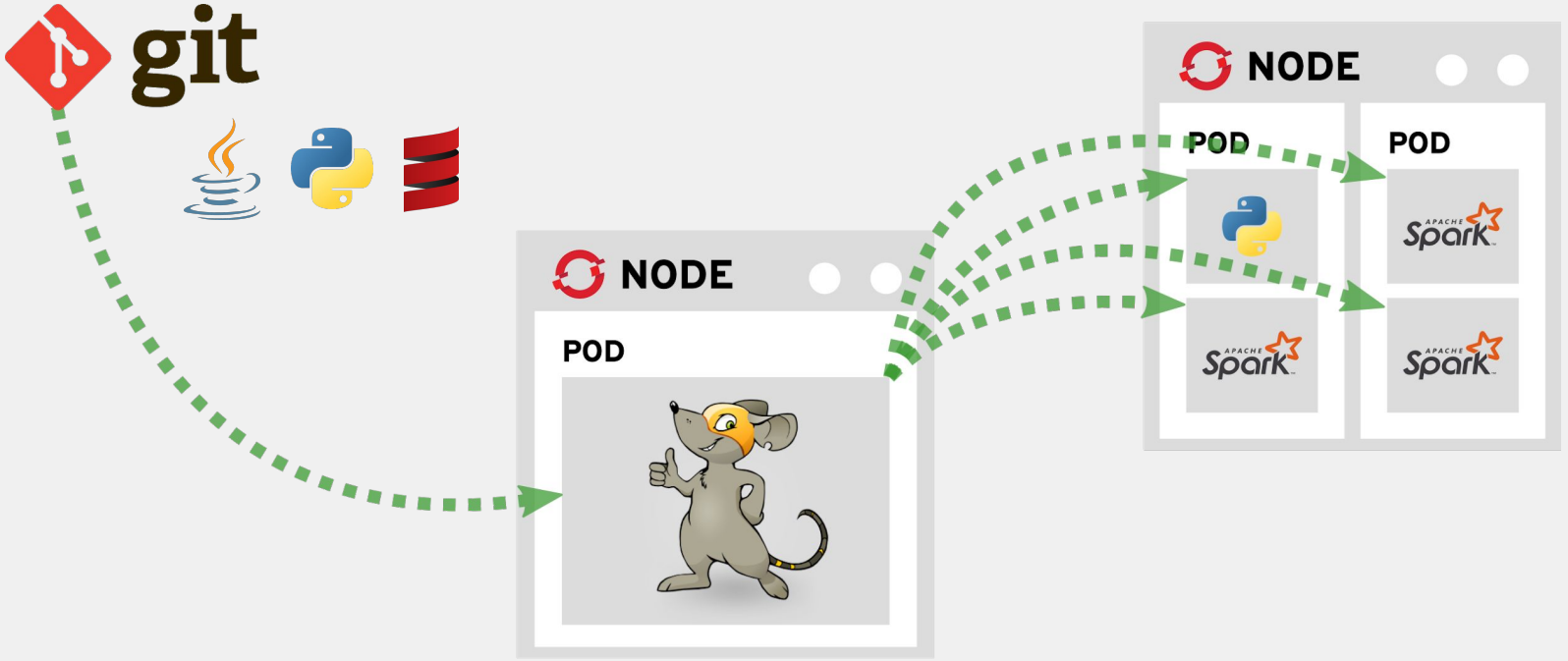
Oshinko deployment



Oshinko deployment



Oshinko deployment



Adding intelligence with Drools

- Business Rules Management System (BRMS)
- visual or text based rules management
- empowers collaboration
- JBoss community

Simple rule example

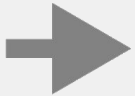
Data Models

Person

```
String name
Date birthday
```

Date

```
Integer day
Integer month
Integer year
```



Rules

```
rule "Happy Birthday"
  when
    $today: Date()
    $p: Person(birthday == $today)

  then
    Print("Happy Birthday " + $p.name);
end
```

Case Study: Financial Fraud Detection

Around 2 million

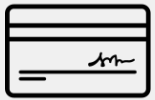
online fraud incidents

were reported in 2017!

User case

“A financial lender would like to deploy a new layer of credit fraud detection by collecting all transactions over a predetermined limit. These transactions will then be processed by an intelligent business rules system to determine evidence of fraud.”

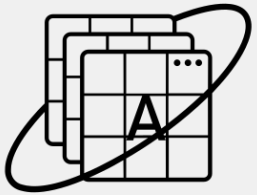
User story

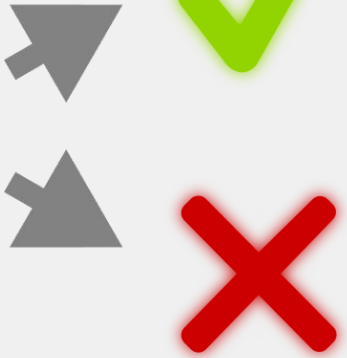


User's details are stolen

Bad guy makes a large transaction

The application identifies this





Is it a fraudulent transaction?

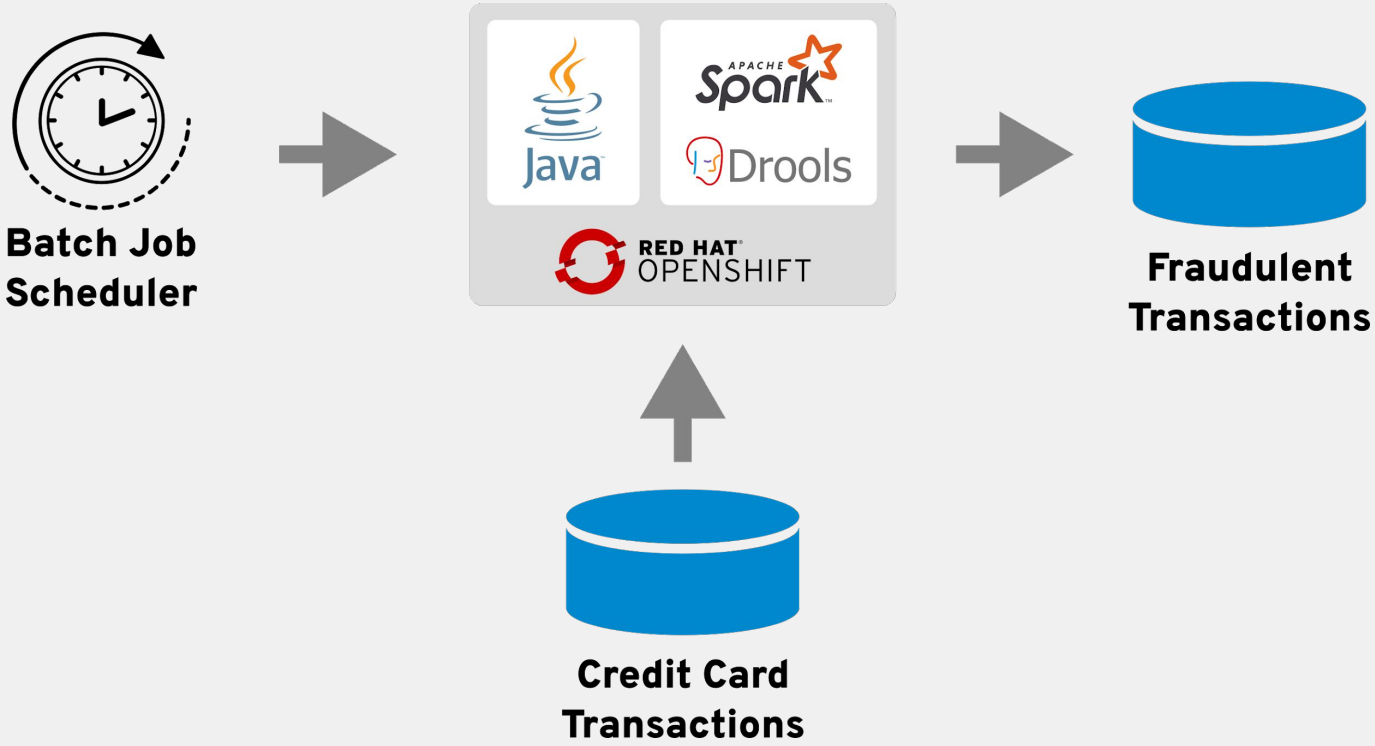
Data set

- Kaggle data set
- credit card transactions
- we will use amount for this example
- is the transaction fraudulent?

Application design

- Spark integrated into a Jupyter Notebook
- separation of concerns: data filtering and rules engine using microservices
- no need for data science expertise with the help of oshinko

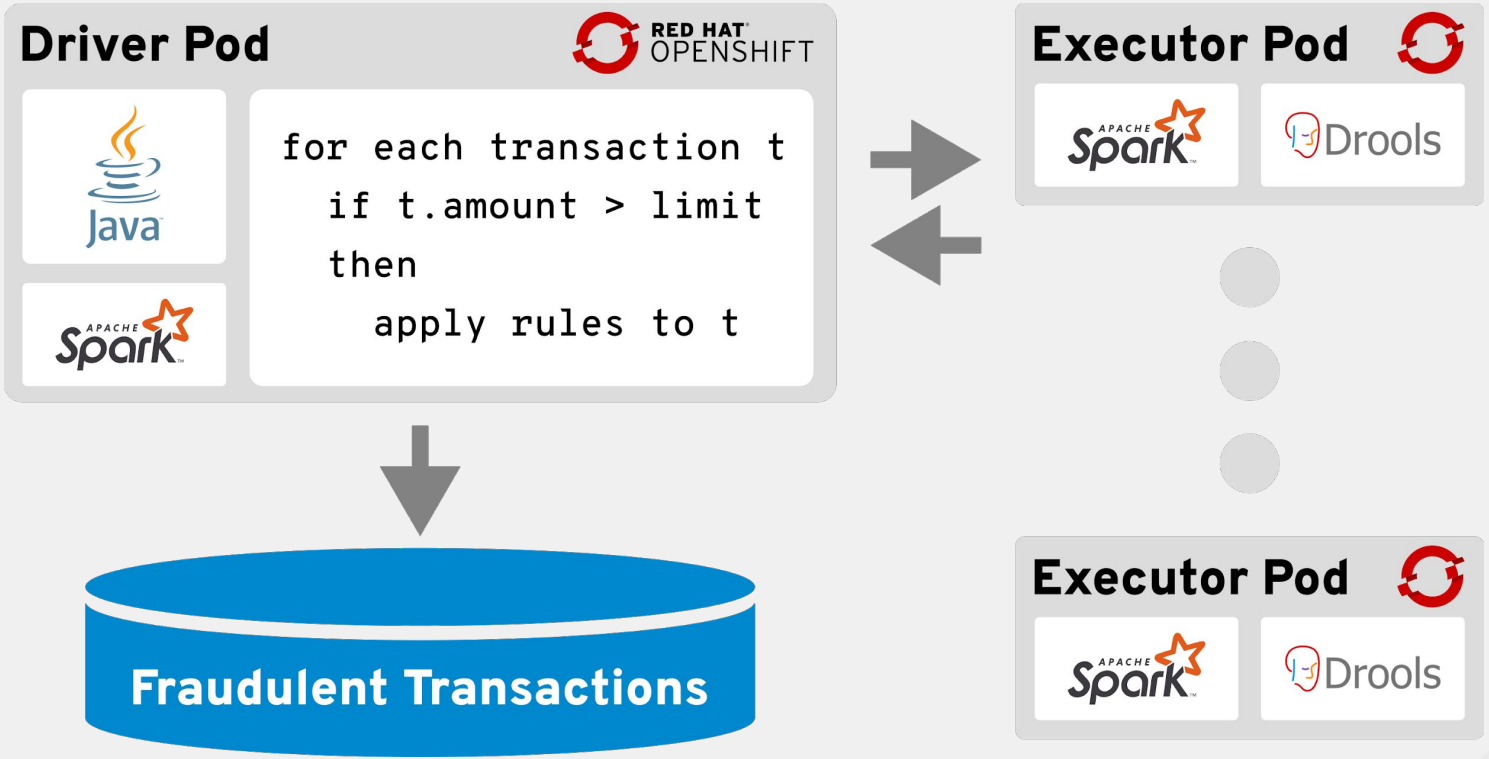
High level architecture



Power of: Spark + Drools

- Spark allows any user to leverage ML without being a data scientist
- separation of concerns between filtering and rule based decisions
- Drools empowers domain experts to share their knowledge and easily make rule based applications

Spark + Drools in this application



Drools rule

```
rule “Detect Fraud”
+ When
+ t:Transaction(v1<0)
+ then:t.setFraudulent(true)
+ end
```

Demo

What’s next for production?

- adding a test configuration
- additional Microservices- e.g. comparing the type of transaction
- migrating Drools jar to separate Maven repository

Lessons learnt

- **good** Communication**!**
- agile design
- distributed and parallel working is **GOOD!**-if you use it responsibly
- clear timeline and deadlines

Have a go yourself it's easy!

- radanalytics.io - GO visit it
- Openshift empowers collaboration
- data science made easy

github.com/rebeccaSimmonds19/transaction_limit

Rebecca Simmonds
rsimmond@redhat.com
@becky_simmonds

Michael McCune
msm@redhat.com
@FOSSjunkie