SparkApplicationClass CRD proposal (Draft)

Summary

Kubernetes Operator for Apache Spark implements *SparkApplication* and *ScheduledSparkApplication* CustomResourceDefinitions (CRDs) to run or schedule spark applications in kubernetes (k8s).

SparkApplicationClass is a proposed k8s CRD to describe classes of spark applications which can be inherited by *SparkApplication* and *ScheduledSparkApplication*. This abstraction layer allows cluster administrators to declare a common set of spark application classes specific to their spark and k8s clusters, and/or their workload use cases.

API Definitions

```
ScheduledSparkApplication
|__ ScheduledSparkApplicationSpec
    |__ SparkApplication
|__ ScheduledSparkApplicationStatus

|__ SparkApplication
|__ SparkApplicationSpec
    |__ DriverSpec
        |__ SparkPodSpec
    |__ ExecutorSpec
        |__ SparkPodSpec
    |__ SparkUISpec
    |__ Dependencies
    |__ MonitoringSpec
        |__ PrometheusSpec
|__ SparkApplicationStatus
    |__ DriverInfo

SparkApplicationClass
|__ SparkApplication
```

Proposed Changes

~~`sparkoperator.k8s.io/class`~~

~~The updated `SparkApplication` and `ScheduledSparkApplication` will support an optional annotation `sparkoperator.k8s.io/class`. This annotation describes the name of the `SparkApplicationClass` custom resource with configurations to be inherited by the spark application.~~

`SparkApplication`

The updated `SparkApplication` will add 2 additional fields:

Field	Type	Note
SparkApplicationClass	String	Name of an existing `SparkApplicationClass` CRD that the `SparkApplication` will inherit from.
SparkUI	`SparkUISpec`	Optionally specifies additional properties for exposing the spark UI with k8s service and ingress.

SparkApplicationClass

A `SparkApplicationClass` has the following top-level field:

Field	Type	Note
Template	`SparkApplication`	A template of `SparkApplication` which describes the default values for a `SparkApplication` or `ScheduleSparkApplication`.

SparkUISpec

A `SparkUISpec` has the following top-level fields:

Field	Type	Note
Host	String	Endpoint to the SparkUI (will be used to configure the Ingress endpoint). Supports the following mustache variables: - `{{$appName}}` - `{{$serviceName}}` - `{{$namespace}}` e.g. `{{$appName}}.ingress.cluster.com` **NOTE:** Available as mustache variable `{{$host}}` in `SparkUISpec.Service` and `SparkUISpec.Ingress`.

Service	`Map[string]string`	An unstructured key value map which can be used by custom api gateways controllers (e.g. ambassador) to gateway to the SparkUI service.
		
		Supports the following mustache variables:
		- `{{$appName}}`
		- `{{$serviceName}}`
		- `{{$namespace}}`
		- `{{$host}}`
		
		e.g
		`sparkUI:`
		`  host: jobs.spark.com`
		`  service:`
		`  getambassador.io/config: |`
		`  ---`
		`  apiVersion: ambassador/v1`
		`  kind: Mapping`
		`  name: ${{appName}}-${{namespace}}`
		`  host: ${{host}}`
		`  prefix: /${{appName}}/`
		`  service: ${{serviceName}}.${namespace}}`

Ingress	`Map[string]string`	An unstructured key value map which can be used by custom ingress controllers (e.g. nginx-ingress-controller) to configure ingresses to the SparkUI service.

		Supports the following mustache variables:
		- {{$appName}}
		- {{$serviceName}}
		- {{$namespace}}
		- {{$host}}
		
		e.g
		`sparkUI:`
		`  host: {{$appName}}.ingress.cluster.com`
		`  ingress:`
		`  kubernetes.io/ingress.class: "nginx"`

Example

`SparkApplication` inherits defaults from `SparkApplicationClass` *pyspark2*.

```
apiVersion: "sparkoperator.k8s.io/v1beta1"
kind: SparkApplication
metadata:
  name: pyspark-pi
spec:
  sparkApplicationClass: pyspark2  # inherits form SparkApplicationClass “pyspark2”
  mainApplicationFile: local:///opt/spark/examples/src/main/python/pi.py
  driver:  # overwrites driver settings
    cores: "500m"
    coreLimit: "1000m"
  executor:  # overwrites executor settings
    cores: 1
    instances: 20
    memory: "1Gi"
```

`SparkApplicationClass` with basic configuration for pyspark (with python 2) and annotations for sparkUI service and ingress.

```
apiVersion: "sparkoperator.k8s.io/v1beta1"
kind: SparkApplicationClass
metadata:
  name: pyspark2
  namespace: sparkoperator
spec:
  template:
    type: Python
    pythonVersion: "2"
    mode: cluster
    image: "gcr.io/spark-operator/spark-py:v2.4.0"
    imagePullPolicy: Always
    sparkVersion: "2.4.0"
    restartPolicy:
      type: OnFailure
      onFailureRetries: 3
      onFailureRetryInterval: 10
      onSubmissionFailureRetries: 5
      onSubmissionFailureRetryInterval: 20
    driver:
      cores: 0.1
      coreLimit: "200m"
      memory: "512m"
      labels:
```

```
    version: 2.4.0
  serviceAccount: spark
executor:
  cores: 1
  instances: 1
  memory: "512m"
  labels:
    version: 2.4.0
sparkUI:  # defines default configs for sparkUI
  host: "${{appName}}.sparkui.com"  # represented as ${{host}}
  service:
    getambassador.io/config: |
      ---
      apiVersion: ambassador/v1
      kind:  Mapping
      name:  ${{appName}}-${{namespace}}
      host: ${{host}}
      prefix: /
      service: ${{serviceName}}.${namespace}}
  ingress:
    kubernetes.io/ingress.class: "nginx"
```