Dataverse Scaling: Sprint 2 Demo

Students: Michael Clifford, Patrick Dillon, Ryan Morano & Ashwin Pillai

Mentors: Phil Durbin (Harvard), Dan McPherson & Solly Ross (both Red Hat)

Work completed

- Researching: Kubernetes by Example, kubernetes.io Stateful Apps tutorial, Postgres images, Docker images.

- Preliminary work in creating StatefulSets
  - Standalone(i.e. not part of DataVerse, pods not initializing)
  - Glassfish (preliminary - image not found error)
  - Postgres(preliminary)

- Updating config/openshift.json on our fork of IQSS/Dataverse for StatefulSets
- Updated schedule

StatefulSets

- Manages the deployment and scaling of a set of Pods, *and provides guarantees about the ordering and uniqueness* of these Pods.

- Like a Deployment, a StatefulSet manages Pods that are based on an identical container spec. Unlike a Deployment, a StatefulSet maintains a sticky identity for each of their Pods. These pods are created from the same spec, but are not interchangeable: each has a persistent identifier that it maintains across any rescheduling.

- A StatefulSet operates under the same pattern as any other Controller. You define your desired state in a StatefulSet object, and the StatefulSet controller makes any necessary updates to get there from the current state.

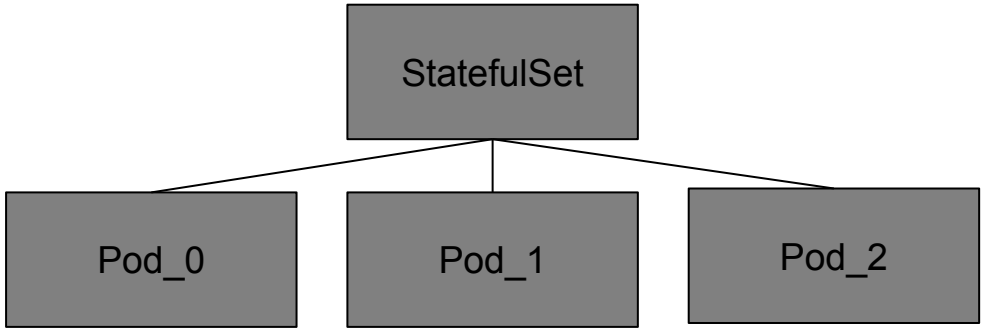
kubernetes

StatefulSets - Components

- A Headless Service, used to control the network domain. Allows a client to get the IP addresses of each pod while preventing automatic interaction with the pods. This prevents clients from interacting with incorrect pods in a StatefulSet
- The StatefulSet itself, which includes the specs for the deployment including the docker images to deploy and the number of replicas of each pod to initialize with.
- The VolumeClaimTemplate provides stable storage for the statefulset. For our testing we will be using EmptyDir

kubernetes

Architecture



Each pod is deployed from the same docker image. However, since all pods after the initial pod need to “know” they are not the original pod, updates to the docker images for each application need to be updated to reflect this relationship

OpenShift Configuration files

General StatefulSet Implementaion:
https://github.com/MichaelClifford/Scaling/blob/master/dverz.json

Dataverse Postgres Implementation:
https://github.com/EC528-Dataverse-Scaling/dataverse/commit/be80972bfc4f16d7b3199723f30d3c06e0476b75

Next Steps

- Finalize Postgres StatefulSets
  - Decide on & configure new postgres image that will work with master/slave
  - Change existing service to headless service

- Finalize Glassfish StatefulSets
  - Modify Glassfish image to enable/disable timer if not master
  - Implement service discovery
  - Modify image to talk to Postgres service and choose appropriate master/slave
  - Determine whether headless service is also necessary for Glassfish

Updated Schedule

Old	New
Release #2 (Feb 22) - PostgreSQL	Release #2 - Begin PostgreSQL & Glassfish StatefulSets
Release #3 (Mar 15) - PostgreSQL	Release #3 - Continue PostgreSQL & Glassfish SS
Release #4 (Mar 29) - Glassfish SS →	Release #4 - Finish PostgreSQL & Glassfish SS
Release #5 (Apr 12) - Glassfish SS	Release #5 - Begin deployment/load test on MOC & work on Solr
Release #6 (Apr 26) - MOC & Solr	Release #6 - Finish deployment/load test on MOC & Solr

Release Planning

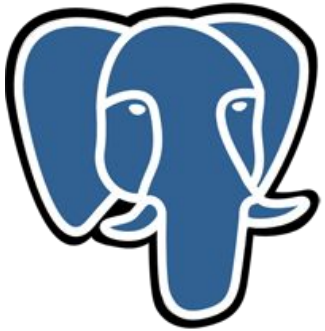
https://tree.taiga.io/project/msdisme-2018-bucs528-template-6/

THANKS!!

The Dataverse Project

Solr

redhat



PostgreSQL