

whois



Kyma Maintainer



Service Catalog SIG Chair

Mateusz Szostok

Senior Software Engineer
at SAP Labs Poland

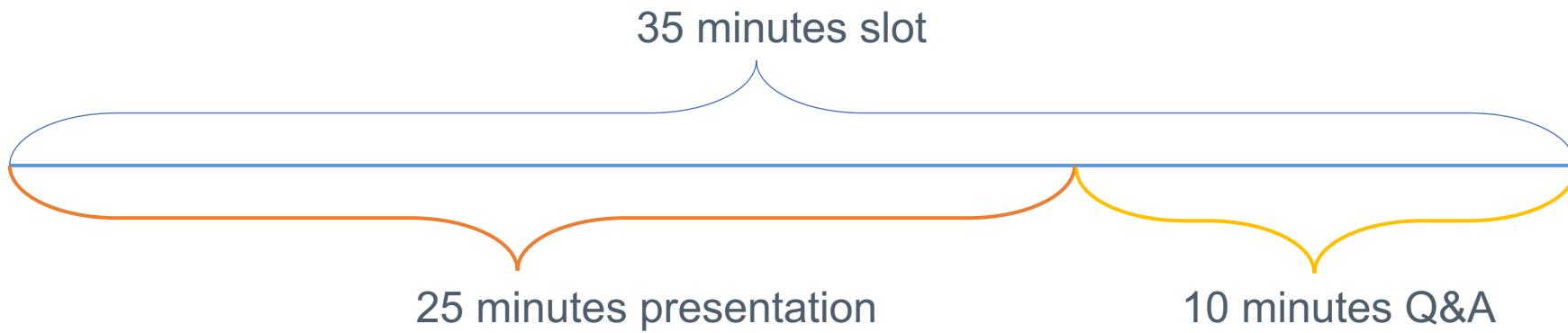


Service Catalog SIG Update

Mateusz Szostok, SAP

Jonathan Berkhahn, IBM

Timeline

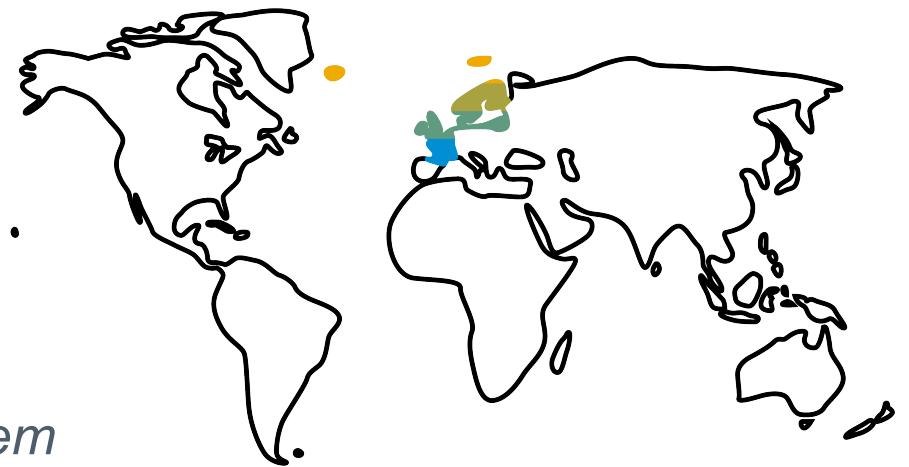


You will learn

- Service Catalog magic a.k.a use-cases
- Open Service Broker API Spec
- Demo
- YAML all the things
- Project updates

Applications are rarely islands

- Often applications leverage ancillary "Services"
e.g. Application stores data in database
- Critical to application's success
But developers shouldn't spend their time managing them



Services - an overloaded term

- Kubernetes “Services”
 - Applications running in the cluster accessible via DNS discovery*
- Platform managed/hosted Services
 - e.g. Object Storage
- External Services - 3rd Party Services
 - e.g. Azure Text Analytics



Creating and accessing services is a challenge

Creating and managing services is non-trivial

- Duplication of effort across teams
- Ops team manages it for you on their schedule
- Managing credentials could be problematic
 - Sent via email, sticky-notes, etc...
 - Where are they stored? Plain text in config files?
- Each service has its own set of provisioning APIs



KEEP
CALM
AND
SUBMIT A
TICKET

What if ...?

```
$ svcat marketplace
```

CLASS	PLANS	DESCRIPTION
mysql	free	Simple SQL
	basic	
	enterprise	
mongodb	free	No-SQL DB

```
$ svcat provision myDB --class mysql --plan free
```

8

```
$ svcat bind myDB
```

Credentials (and connection info) in “myDB” secret

The magic

Cluster Admin:

- **Service Brokers** are registered with Kubernetes
 - Each Broker manages one or more **Services**
 - Each Service offers a set of variant-QoSs/**Plans**
- Services are available via the “**Marketplace**” in Kubernetes

`$ svcat marketplace`

Developer:

- Chooses a **Service** from the **Marketplace**
- Kubernetes talks to owning **Broker** to provision it and obtain the credentials
- **Secret** (credentials, connection info) is available to the app

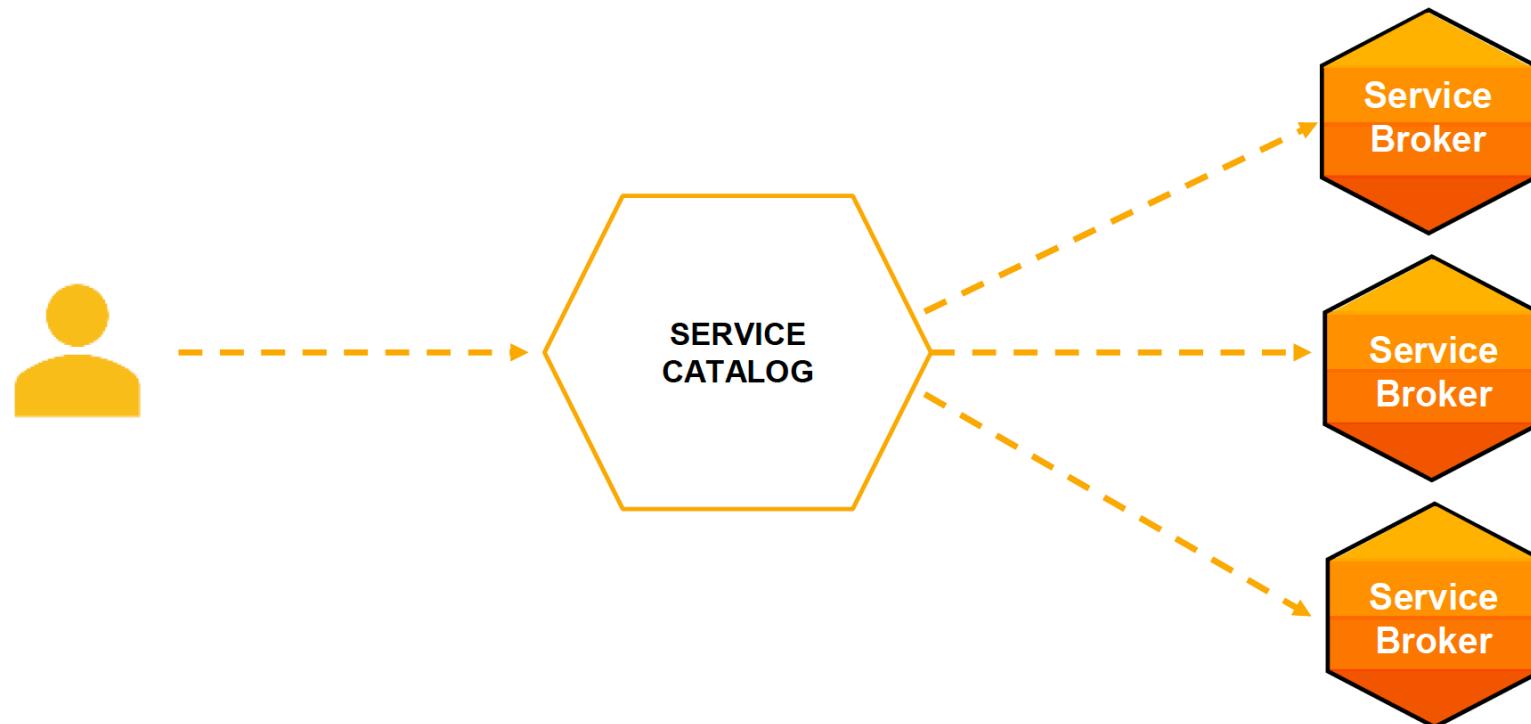
`$ svcat provision`

`$ svcat bind myDBd`



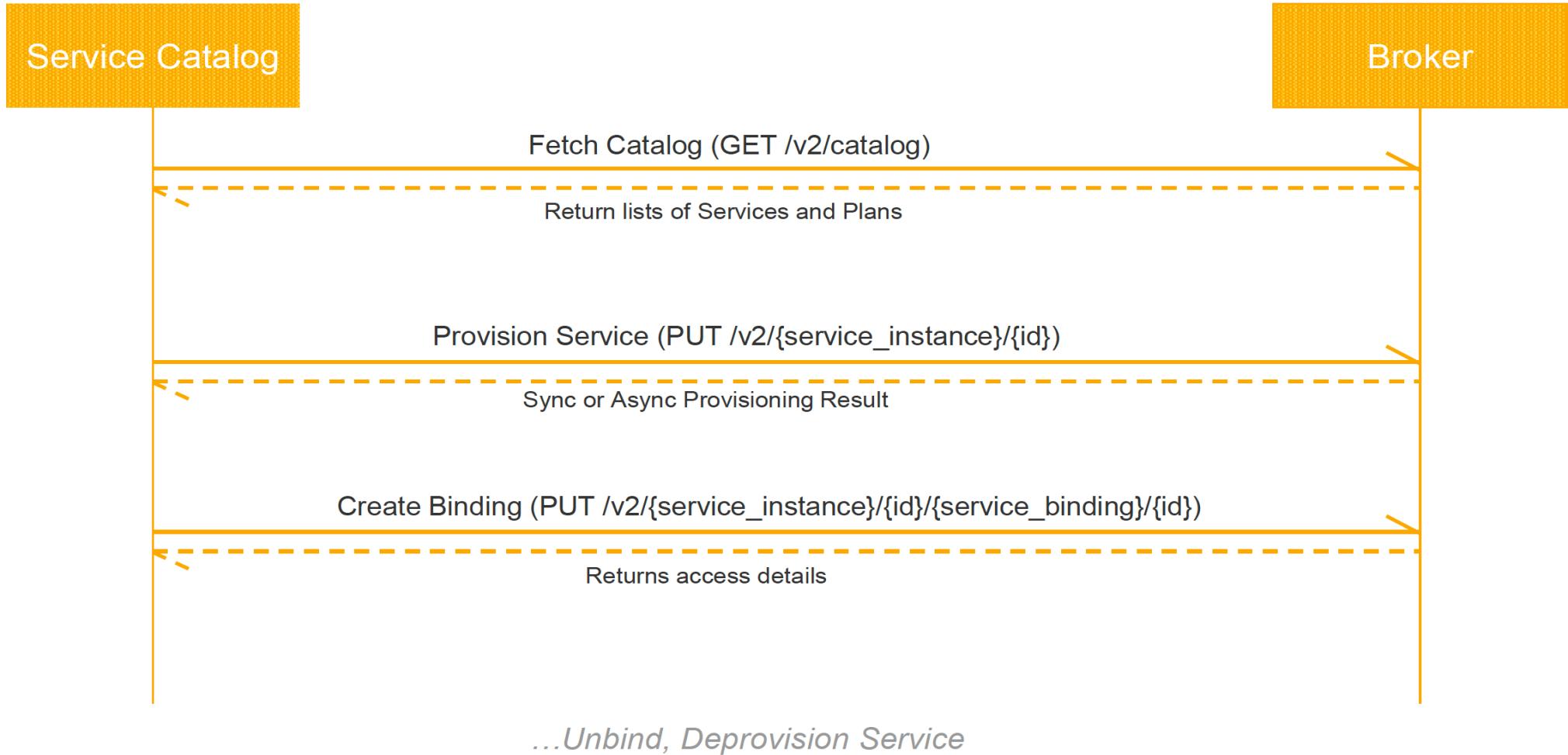
Open Service Broker API Spec

The Open Service Broker API defines the interaction between the platform and a broker





Open Service Broker API



Demo



YAML all the things

```
apiVersion: servicecatalog.k8s.io/v1beta1
kind: ServiceInstance
metadata:
  name: myDB
spec:
  serviceClassName: mysql
  planName: free
```

```
apiVersion: servicecatalog.k8s.io/v1beta1
kind: ServiceBinding
metadata:
  name: myDB
spec:
  instanceRef:
    name: myDB
```

**Credentials and connection info in “myDB” secret*



Service Catalog

- **ClusterServiceBroker** (and namespaced **ServiceBroker**)
A server running somewhere that offers various services, e.g. MySQL Broker
- **ClusterServiceClass** (and namespaced **ServiceClass**)
A category of services offered by a Broker, e.g. MySQL Databases
- **ClusterServicePlan** (and namespaced **ServicePlan**)
A specific type of a Service that a Broker offers, e.g. 100 MB MySQL Databases
- **ServiceInstance**
A single instantiation of a Service/Plan, e.g. Matt's 100 MB MySQL Database
- **ServiceBinding**
A unique set of creds to access a specific Instance, e.g. username/password for Matt's 100 MB MySQL Database



Service Catalog

- **ClusterServiceBroker** (and namespaced **ServiceBroker**)

A server running somewhere that offers various services, e.g. MySQL Broker

```
apiVersion: servicecatalog.k8s.io/v1beta1
kind: ClusterServiceBroker
metadata:
  name: sample-broker
spec:
  authInfo:
    basic:
      secretRef:
        name: sample-broker-auth
        namespace: brokers
  catalogRestrictions:
    serviceClass:
      - "spec.externalName in (FooService, BarService)"
  url: http://sample-broker.brokers.svc.cluster.local
```

Service Catalog Summary

Why?

- Helps developers discover and connect to 3rd party services
- Allows you to focus on business logic
 - Ask for the service
 - Connection information provided at runtime
- Services managed by the experts

Status

- Kubernetes sigs project
- Can be deployed into any Kubernetes cluster via a Helm chart
- Beta version

Updates





Mateusz Szostok
@m_szostok

Obserwuj

@kubernetesio guess what? After a few months of work, the new Kubernetes Service Catalog release is now available!

The Aggregated API Server is a thing of the past. Now the CRDs and Admission Webhooks are the main players in the house!



New architecture in place!

#bragging #ServiceCatalog #Kubernetes



01:25 - 3 paź 2019

5 podań dalej 14 polubień



New architecture in place!

Latest release

v0.3.0

11c5b36

Verified

Compare ▾

v0.3.0

mszostok released this on May 25 · 6 commits to master since this release

Service Catalog v0.3.0 is now available!

This is a **MAJOR** release. In release 0.3.0, we've focused on replacing the Aggregated API Server with the CustomResourceDefinitions (CRDs) and the Admission Webhook solution. The [Service Catalog documentation](#) has been updated to reflect these changes. Check it out for more details.

To install the Service Catalog from this release, run:

```
helm install svc-cat/catalog \  
--name catalog --namespace catalog --version 0.3.0
```

If you installed Service Catalog on your cluster using the `helm install svc-cat/catalog` command, then the migration process is automated for you and you can just run:

```
helm upgrade <release_name> svc-cat/catalog --version 0.3.0
```

The migration process is described in detail [here](#). There you will also find information on how to upgrade the Service Catalog manually.

Highlights of the changes since 0.3.0-beta.2

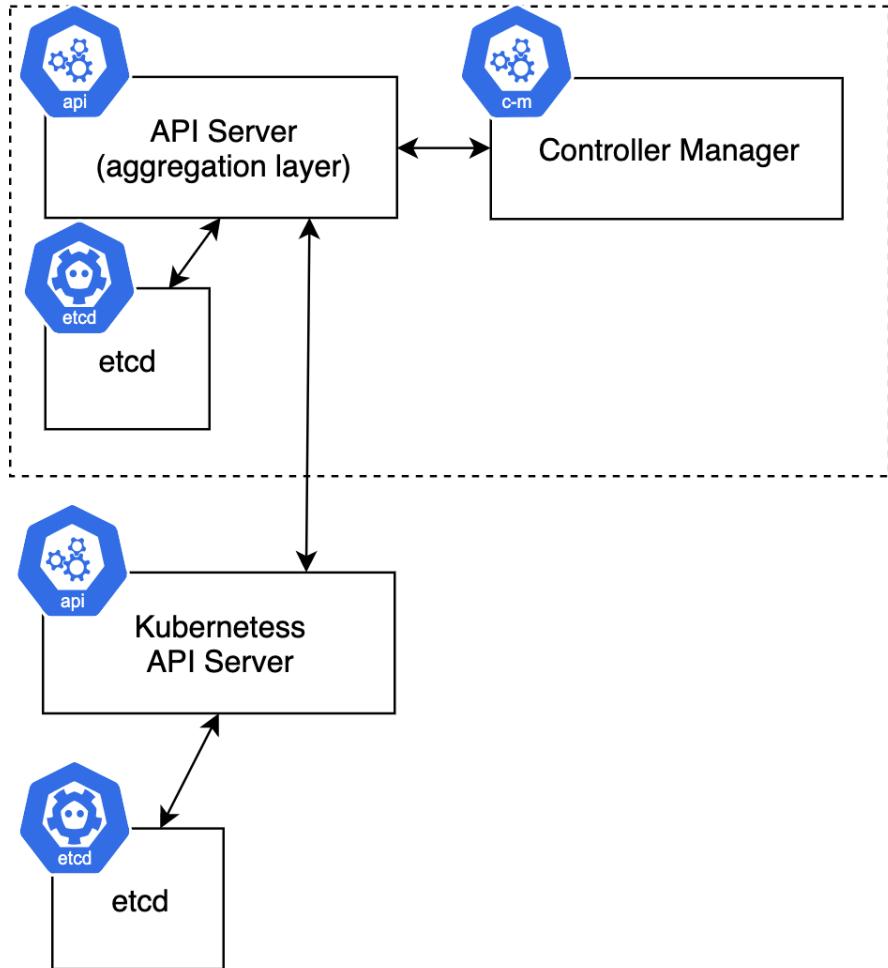
- Cascading binding deletion feature. The controller manager deletes all Service Bindings for a Service Instance before attempting to deprovision the Service Instance. This option can be enabled by setting the [CascadingDeletion feature gate](#) to true ([#2711](#))
- To perform a successful migration, the Service Catalog resources can't be in the unfinished or deleted state. Otherwise, the upgrading job can fail. We provide a special sanity check script that should be executed before upgrading from 0.2.x to 0.3.0 ([#2743](#))
- Previously, when (Cluster)ServicePlan was marked as `removedFromBrokerCatalog`, it would not be removed on the (Cluster)ServicePlan reconcile due to a bug that blocked the removing logic. This bug was fixed in this release. ([#2797](#))
- There are situations when Service Instance and Service Binding are applied at the same time (used e.g. in automation scripts). In those cases, usually, ServiceBinding will fail if ServiceInstance will not be ready in time. In this release, there is a new functionality to trigger reconciliation for each ServiceBinding with failed status if correlated ServiceInstance was finally provisioned successfully. ([#2771](#))
- svcat CLI can now filter the classes returned by a broker in `svcat get classes` via a `--broker` flag ([#2786](#))
- Support for Kubernetes 1.18 and matrix tests for validating Service Catalog on the three latest Kubernetes versions ([#2796](#))

For the rest of the merge pull request, check the [All changes since beta.2](#) section.

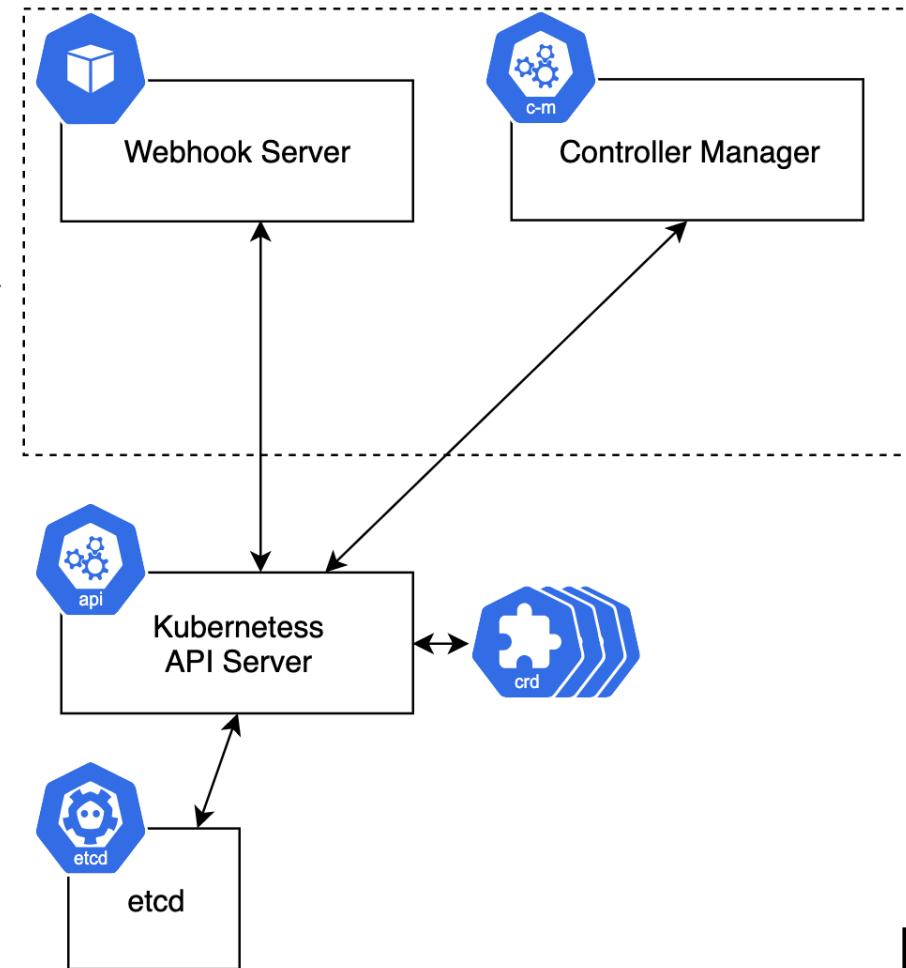
We'd appreciate any feedback on the upgrade procedure and any issues or tips you may run into.

New architecture

Service Catalog 0.2.0



Service Catalog 0.3.0



migration



How these plans affect you

master (v0.3 - CRDs)

```
helm install svc-cat/catalog \
--name catalog --namespace catalog
```

```
helm upgrade catalog svc-cat/catalog
```

v0.2 (api-server)

```
helm install svc-cat/catalog-v0.2 \
--name catalog --namespace catalog
```

```
helm upgrade catalog svc-cat/catalog-v0.2
```

Not supported since July 2020

Subprojects

Minibroker is an implementation of the OSB API suited for local development and testing.

We use it in the official Service Catalog walkthrough documentation.

Repository: <https://github.com/kubernetes-sigs/minibroker>

Go Open Service Broker Client is a Go client for communicating with service brokers which implements the OSB API.

We use it in the Service Catalog controllers.

Repository: <https://github.com/kubernetes-sigs/go-open-service-broker-client>

Next plans – milestone 0.3.1

- Documentation enhancements (compliance with official guidelines, doc structure clean-up , etc.)
- Compliance with the new [Open Service Broker API 2.15](#)
- CI pipelines clean-up (probably get rid of Travis and use only Prow)
- Migration of Service Catalog resources under the SIG control
 - <https://svcat.io> → <https://svcat.sigs.k8s.io>
 - <https://download.svcat.sh> → <https://download.svcat.sigs.k8s.io> (*final URL may differ*)
- Decision on the future of the PodPresets functionality

Generic Extensions #2785

open

Samze opened this issue on Mar 20 · 2 comments



Samze commented on Mar 20 · edited ·

Contributor · ...

Feature: Generic Extensions

The OSB API specification has a pending PR to support Generic Extensions.

I'm creating this issue to start a conversation on how this might be implemented in service-catalog and to get feedback on ideas.

Why

From the PR:

The specification defines endpoints that allow the lifecycle management of service instances and service bindings. However, a common complaint is that it does not support some of those important "day 2" operations that developers might want, e.g. backup and restore. Also that it does not allow service specific operations, e.g. MySQL set leader. To accomplish this Service Brokers authors have the option to either go off-spec or to misuse the spec (e.g. cf update-service -p '{"trigger-backup": true}').

The specification needs an extension mechanism to allow authors to define new endpoints.

What is it

Service Brokers can provide OpenAPI documents attached to Plans in their Catalog. The extensions can then be triggered by platforms on defined paths on the Service Broker.

For example a Broker may host the following OpenAPI document to perform a backup:

```
openapi: "3.0.0"
info:
  version: 1.0.0
  title: My Service Extension
paths:
  /backup:
    post:
      summary: Backup of a Service Instance
      operationId: backup
      tags:
        - backup
      responses:
        '202':
          description: Backup accepted
```

And this can be triggered on the broker on a particular path:

```
/v2/service_instances/:instance_id/extensions/broker-extension-path/backup .
```

An example broker implementation can be found [here](#).



General Info

Become a Service Catalog contributor! 😊

Regular SIG Meeting: Mondays at 13:00 PT (Pacific Time) (bi-weekly).
Convert to your timezone.

General Info

- **Chairs**
 - Jonathan Berkhahn - jaberkha@us.ibm.com - [@jberkhahn](https://twitter.com/@jberkhahn)
 - Mateusz Szostok - mateusz.szostok@sap.com - [@mszostok](https://twitter.com/@mszostok)
- **Home page:** svc-cat.io
- **GitHub repo:** github.com/kubernetes-sigs/service-catalog
- **Slack channel:** kubernetes.slack.com/messages/sig-service-catalog
- **Mail List:** groups.google.com/forum/#!forum/kubernetes-sig-service-catalog
- **OSB API Spec:** openservicebrokerapi.org

Contact information



Mateusz Szostok

Senior Software Engineer
at SAP Labs Poland



Slack

slack.k8s.io



Slack

slack.kyma-project.io



Github

github.com/mszostok



Twitter

twitter.com/m_szostok



LinkedIn

linkedin.com/in/mszostok