ACEv11.0.0.2 On

IBM Cloud Private 3.1

Devops driven integration to micro services principles

V0.1 Draft

Dave Arnold

Do Nguyen

Peter Jessup

December 17, 2018

Table of Contents

[Overview - Diagram 4](#_Toc532812260)

[Overview - Description 4](#_Toc532812261)

[The Integration Micro Services 4](#_Toc532812262)

[The docker images 4](#_Toc532812263)

[The Personas 4](#_Toc532812264)

[The Scenario 5](#_Toc532812265)

[Customized Jenkins Pipelines 5](#_Toc532812266)

[Summary of Github Repositories 6](#_Toc532812267)

[Summary of Dockerhub Images 6](#_Toc532812268)

[ACE Integration Liveliness Probe 7](#_Toc532812269)

[Testing Liveliness Probe 7](#_Toc532812270)

[Github Source Repos for ACE Liveliness Probe (The SoE ACE project) 8](#_Toc532812271)

[Building the Image for SoE – GitHub ACE SoE Build 8](#_Toc532812272)

[Microclimate SOE Project 8](#_Toc532812273)

[Jenkins Pipeline for SoE 9](#_Toc532812274)

[SoE Image on ICP 9](#_Toc532812275)

[Deploying the SoE 9](#_Toc532812276)

[ACE Integration Micro Service 1 10](#_Toc532812277)

[Testing Micro Service 1 10](#_Toc532812278)

[Github Source Repos for ACE Micro Service 1 11](#_Toc532812279)

[Building the Image for Micro Service 1 – GitHub ACE Micro Service 1 Build 11](#_Toc532812280)

[Microclimate Micro Service 1 Project 12](#_Toc532812281)

[Jenkins Pipeline for Micro Service 1 13](#_Toc532812282)

[Micro Service 1 Image on ICP 13](#_Toc532812283)

[Deploying Micro Service 1 on it’s own for testing – Use github Repos ACEonICPIntMicSrvHelm 14](#_Toc532812284)

[Microclimate Project 16](#_Toc532812285)

[Jenkins Pipeline 16](#_Toc532812286)

[ACE Integration Micro Service 2 17](#_Toc532812287)

[Testing Micro Service 2 17](#_Toc532812288)

[Github Source Repos for ACE Micro Service 2 18](#_Toc532812289)

[Building the Image for Micro Service 2 – GitHub ACE Micro Service 1 Build 18](#_Toc532812290)

[Microclimate Micro Service 2 Project 19](#_Toc532812291)

[Jenkins Pipeline for Micro Service 2 20](#_Toc532812292)

[Micro Service 2 Image on ICP 20](#_Toc532812293)

[Deploying Micro Service 2 on it’s own for testing – Use github Repos ACEonICPIntMicSrvHelm 21](#_Toc532812294)

[Microclimate Project 23](#_Toc532812295)

[Jenkins Pipeline 23](#_Toc532812296)

[Deploying Micro Service 1 and Micro Service 2 together as a Micro Services Appliction– Use github Repos ACEonICPIntMicSrvApplHelm 25](#_Toc532812297)

[Integration Micro Services Application Helm Charts GitHub Repository 25](#_Toc532812298)

[Chart Files for Integration Micro Service 1 25](#_Toc532812299)

[Sub Chart files for Integration Micro Service 2 26](#_Toc532812300)

[YAML Chart file modifications - How it hangs together 27](#_Toc532812301)

[Microclimate project aceonicpintmicsrvapplhelm 29](#_Toc532812302)

[Jenkins Pipeline - aceonicpintmicsrvapplhelm 29](#_Toc532812303)

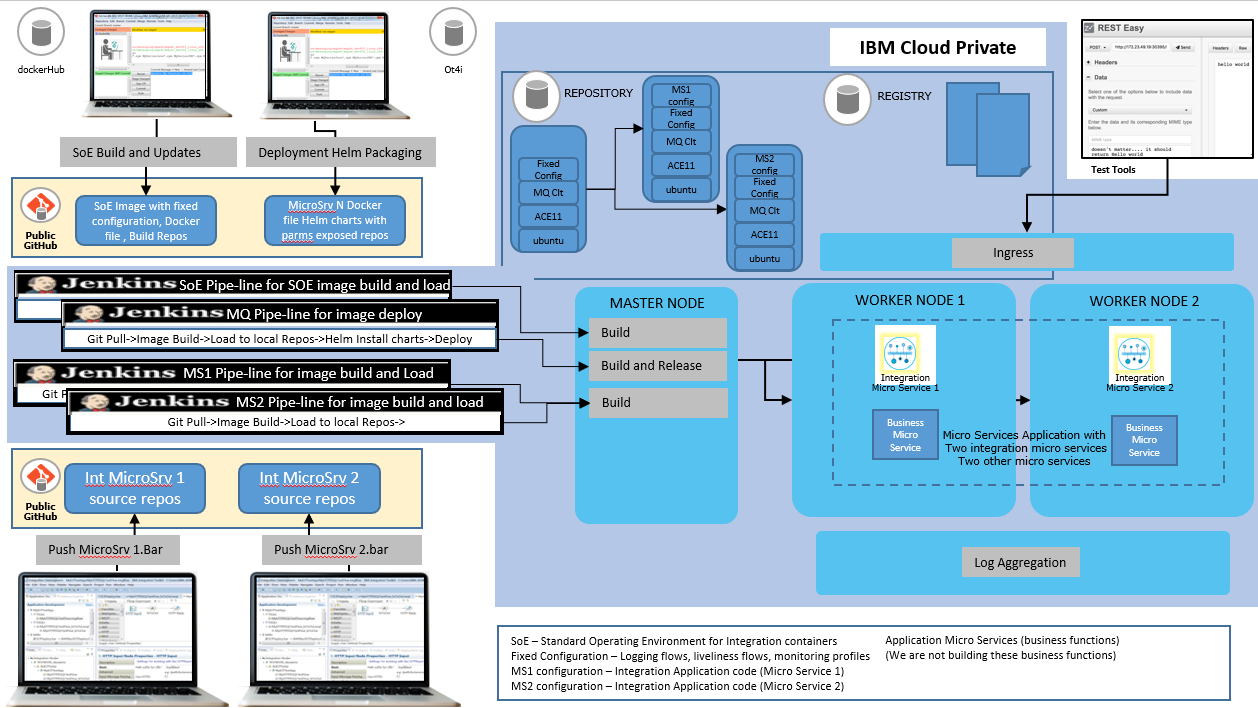
[Helm Release on ICP for Integration Micro Services Application 31](#_Toc532812304)

[Integration Micro Service One 33](#_Toc532812305)

[Integration Micro Service Two 35](#_Toc532812306)

[Testing the Integration Micro Services Application 37](#_Toc532812307)

## Overview - Diagram



## Overview - Description

### The Integration Micro Services

The Integration Micro Service 1 will call Integration Micro Service 2

Integration Micro Service 1 RESTInput(HTTP) ->Mapping Node ->RESTRequest (call Integration Microservice 2)->RESTReply(HTTP)

Integration Micro Service 2 RESTInput(HTTP)->Mapping Node Payload+"Hello from Integration Microservice 2"->RESTReply(HTTP)

Integration Micro Service 2 can be called directly

We have set up the integration in this way in order to position for using Istio as a service mesh to route between integration micro services. However, in this initial example I will make use of the ICP DNS service that uses services names to resolve host names and port numbers. So the base URL on the RESTRequest Node in Micro Service 1 will call Micro Service 2 via its service name.

### The docker images

The ACE Standard operating image is:

Ubuntu

ACEv11.0.0.2

MQ v9.1 Client

Fixed configuration - a bar with RESTful Service, a custom Liveliness Listener

The ACE micro services images build FROM this (base) Standard Operating Environment

### The Personas

Developer 1 and Developer 2 have their own ACE Toolkits and workspaces and are “just” developers. They write ACE services and flows and build BAR files.

Builder(s) are responsible for maintaining the SoE image and building integration microservices images FROM the SoE based on the BAR files created by the Developers

Deployer(s) are responsible for packaging and deployment of micro services application that can include integration micro services. They use Helm and the ICP Devops tooling and pipelines to deploy.

### The Scenario

The ICP Cloud represents an environment that exists in the standard Software Delivery Life Cycle (could be SIT, QA, Perf, Prod (or potentially one of those environments in an ICP cluster namespace)

We are delivering a Micro Services application consisting of multiple micro services two of which have requirements that are integration centric and ACE has been chosen as the runtime for these functions

Our fictitious organization maintains a standard operating environment (container image) for micro service integration with a fixed configuration component on top of which the integration micro services are added and then the images deployed and run as a helm release.

The primary mechanism for deployment will be Jenkins but we'll use Microclimate as it sorts out security on the ICP instance for us and standup the Jenkins toolchains.

## Customized Jenkins Pipelines

Microclimate you have to customize the jenkins build scripts (micro climate is really set up for Java and NodeJS).

We made changes to the Jenkins pipeline scripts in order to get greater flexibility in the Image Build, Deploy Only and Image Build and Deploy. This allowed us greater control in demoing.

<https://github.com/cloudnativedemo/icp-notes/blob/master/microclimate_notes.md>

## Summary of Github Repositories

ACE Developer Project source for a custom Liveliness probe that will deploy into the Standard Operating Image on which all other images are based

<https://github.com/DAVEXACOM/ACEonICPIntMicSoE>

ACE Developer Project source for Integration Micro Service 1

<https://github.com/DAVEXACOM/ACEonICPIntMicSrv1>

ACE Developer Project source for Integration Micro Service 2

<https://github.com/DAVEXACOM/ACEonICPIntMicSrv2>

ACE Standard Operating Environment Image Build - The base image build with Liveliness probe that other images are build FROM

<https://github.com/DAVEXACOM/ACEonICPIntStdOpImg>

ACE Micro Service 1 Image Build

<https://github.com/DAVEXACOM/ACEonICPIntMicSrv1Img>

ACE Micro Service 2 Image Build

[https://github.com/DAVEXACOM/ACEonICPIntMicSrv2Img](https://github.com/DAVEXACOM/ACEonICPIntMicSrv1Img)

Helm Charts that can deploy ACE Micro Service 1 or 2 as an individual for testing (modify Jenkins file and Values.yaml)

<https://github.com/DAVEXACOM/ACEonICPIntMicSrvHelm>

Helm Charts that deploy both ACE Micro Service 1 and 2 as a micro services application

[https://github.com/DAVEXACOM/ACEonICPIntMicSrvApplHelm](https://github.com/DAVEXACOM/ACEonICPIntMicSrvHelm)

## Summary of Dockerhub Images

ACE Developer Project source for a custom Liveliness probe that will deploy into the Standard Operating Image on which all other images are based

<https://cloud.docker.com/u/davexacom/repository/docker/davexacom/ace11002mqc91soe>

ACE Developer Project source for Integration Micro Service 1

<https://cloud.docker.com/u/davexacom/repository/docker/davexacom/ace11002mqc91intms1>

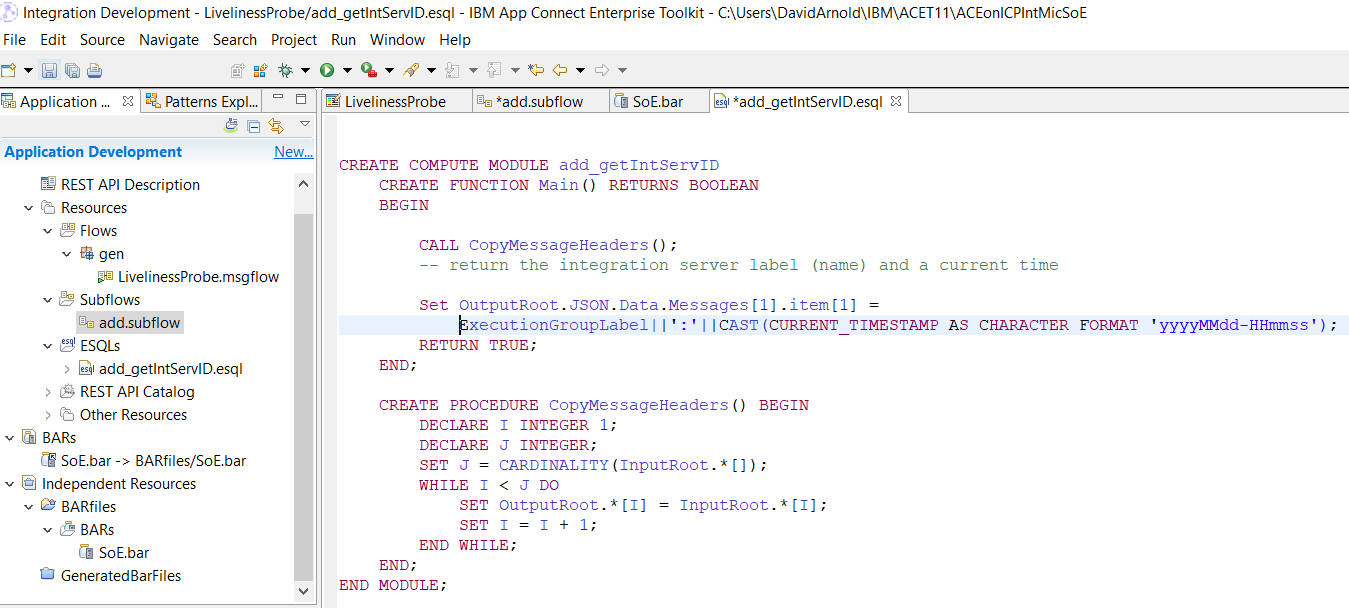
ACE Developer Project source for Integration Micro Service 2

https://cloud.docker.com/u/davexacom/repository/docker/davexacom/ace11002mqc91intms2

You can use the dockerHub images with ICP rather than the on board repository OR pull them to you workstations and play around with them in a docker only (non K8s) environment.

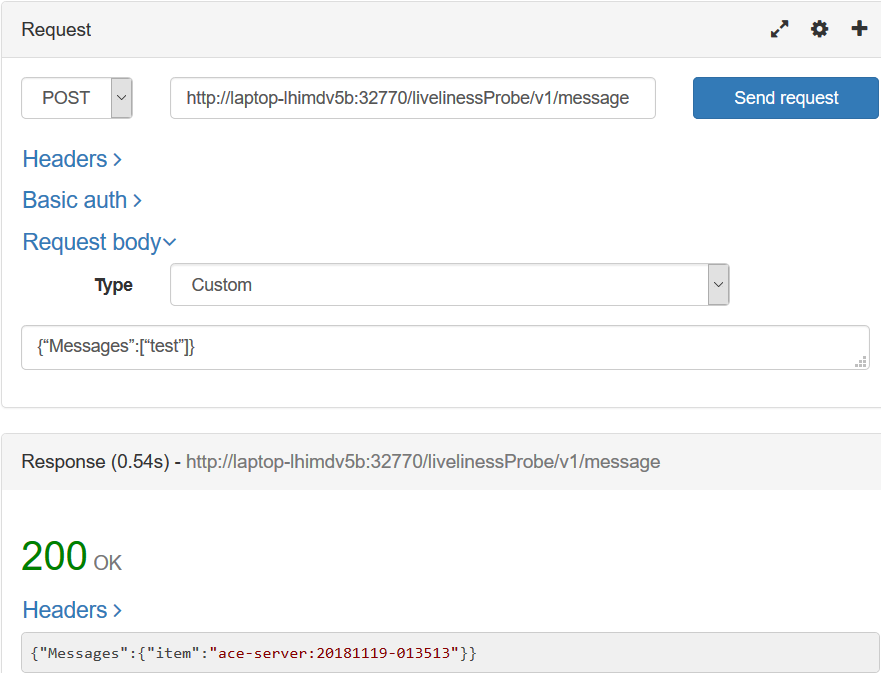
## ACE Integration Liveliness Probe

The Liveliness Probe Service is an other restful service that we deploy into the ACE Standard operating environment. The base image from which Micro Service 1 and Micro Service 2 images are built. So it appears in all ACE Integration Micro Services. (this is not the service baked into the cloud paks we turn those off to demonstrate having a customer centric base standard image. (an SoE).



### Testing Liveliness Probe

Note it returns the integration server name plus a current timestamp



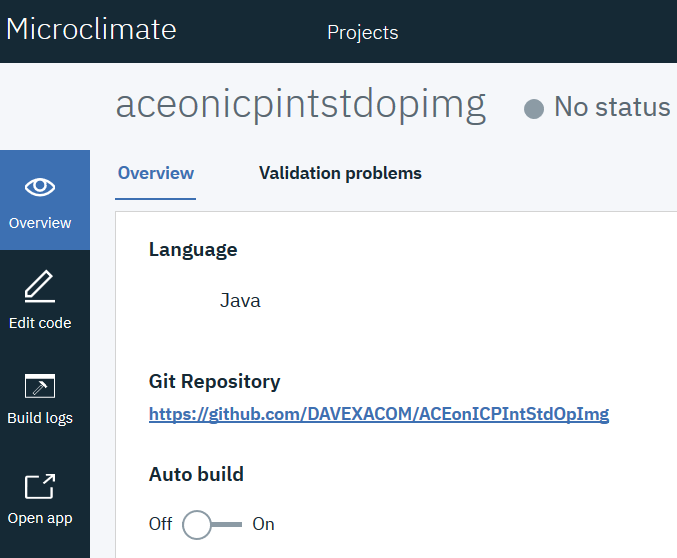
### Github Source Repos for ACE Liveliness Probe (The SoE ACE project)

<https://github.com/DAVEXACOM/ACEonICPIntMicSoE>

### Building the Image for SoE – GitHub ACE SoE Build

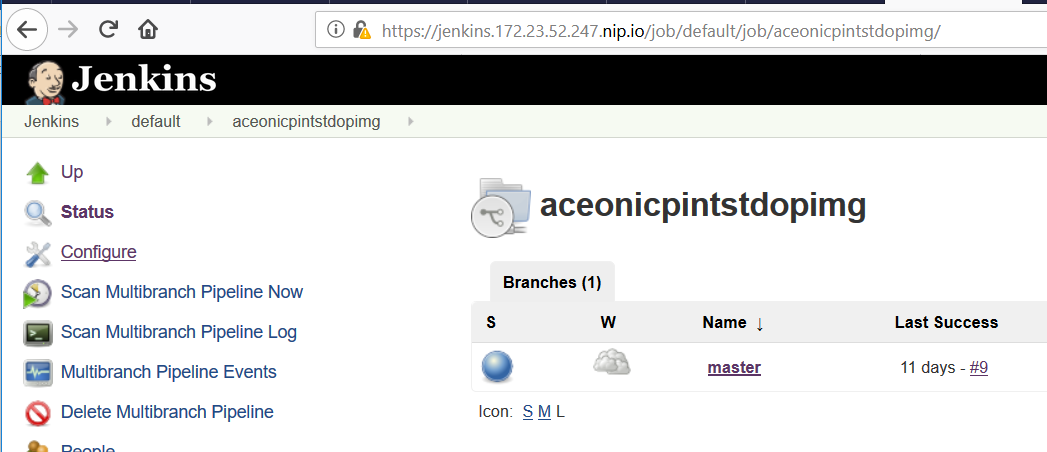
<https://github.com/DAVEXACOM/ACEonICPIntStdOpImg>

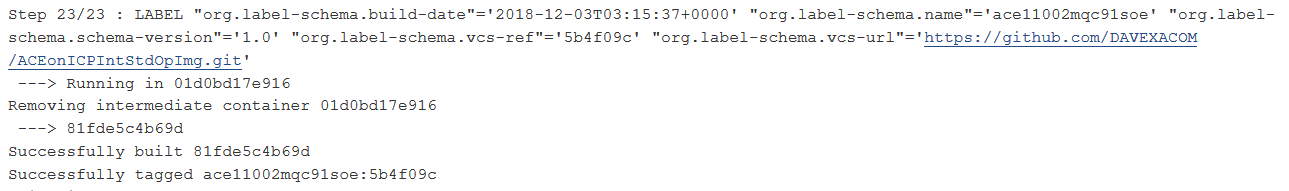
### Microclimate SOE Project



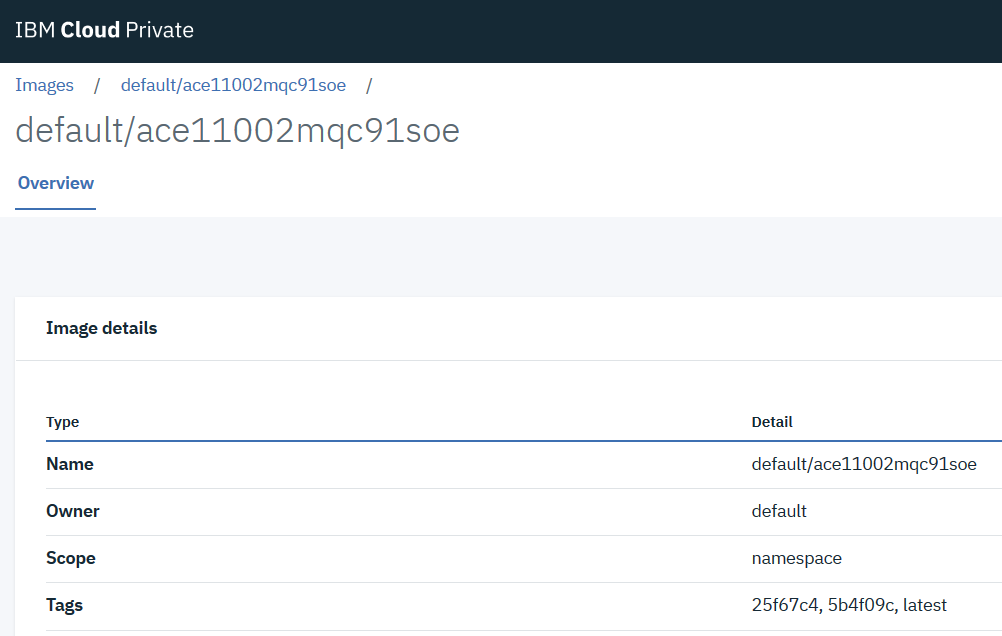


### Jenkins Pipeline for SoE





### SoE Image on ICP

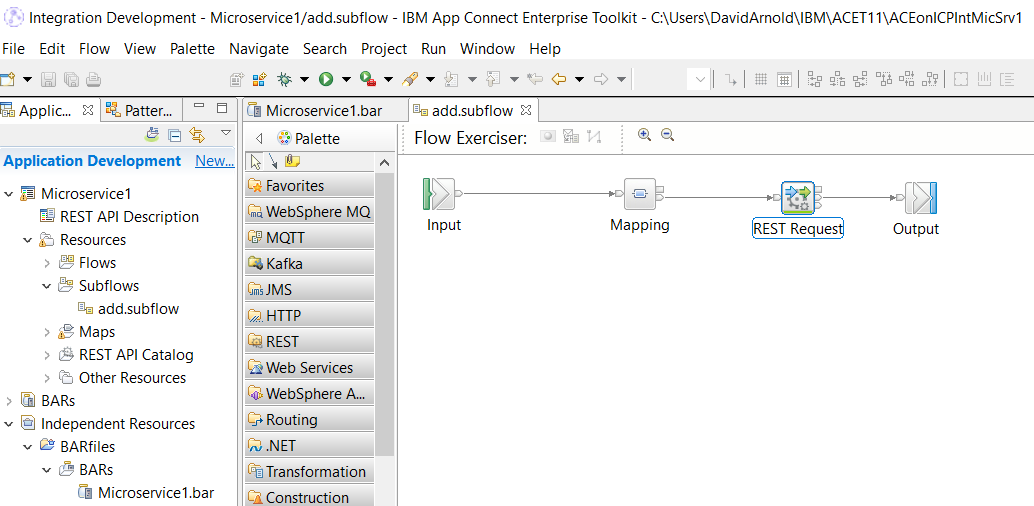


### Deploying the SoE

The SoE Image is never deployed it is the image that Micro Service 1 and 2 are build FROM.

## ACE Integration Micro Service 1

Integration Micro Service 1 calls Integration Micro Service 2 via Rest request

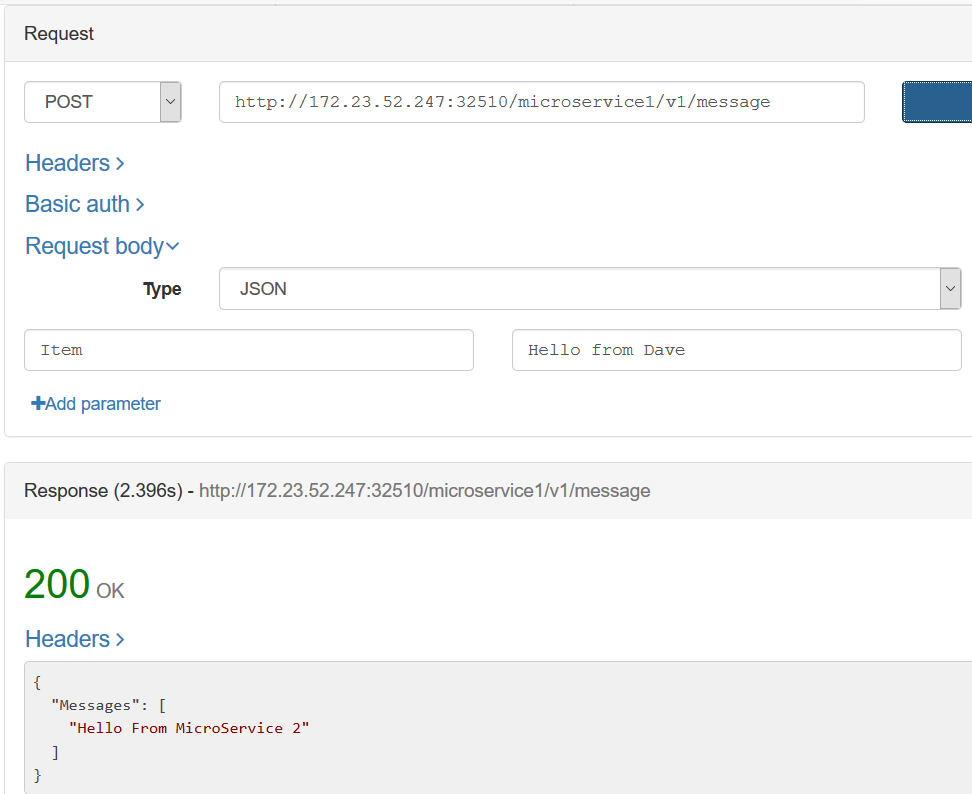


The base URL on the REST Request Node leverages the ICP DNS service naming convention to call MS2 via its service name. This will need to be replaced with Istio service mesh routing.

<http://ace11002mqc91intmsall-intmicsrvtwo:7800/microservice2/v1>

### Testing Micro Service 1

Note it returns hello from Micro Service 2



### Github Source Repos for ACE Micro Service 1

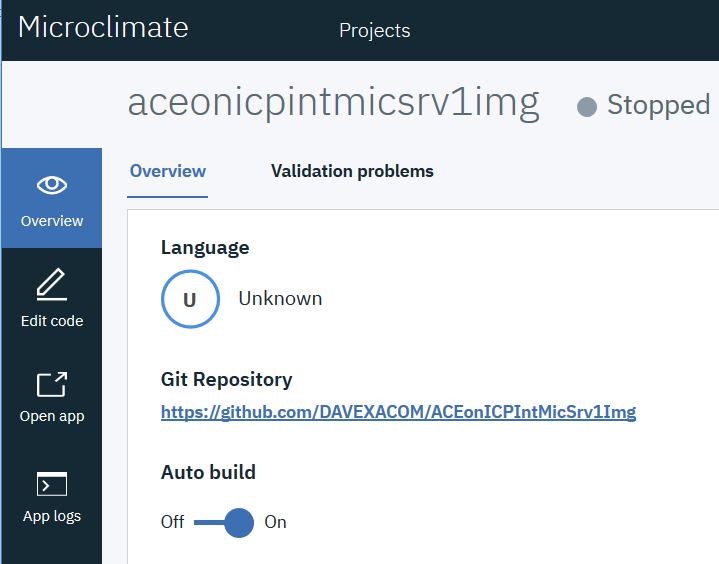
https://github.com/DAVEXACOM/ACEonICPIntMicSrv1

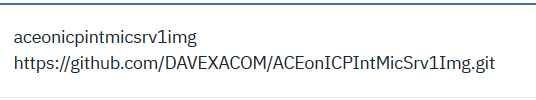
### Building the Image for Micro Service 1 – GitHub ACE Micro Service 1 Build

<https://github.com/DAVEXACOM/ACEonICPIntMicSrv1Img>

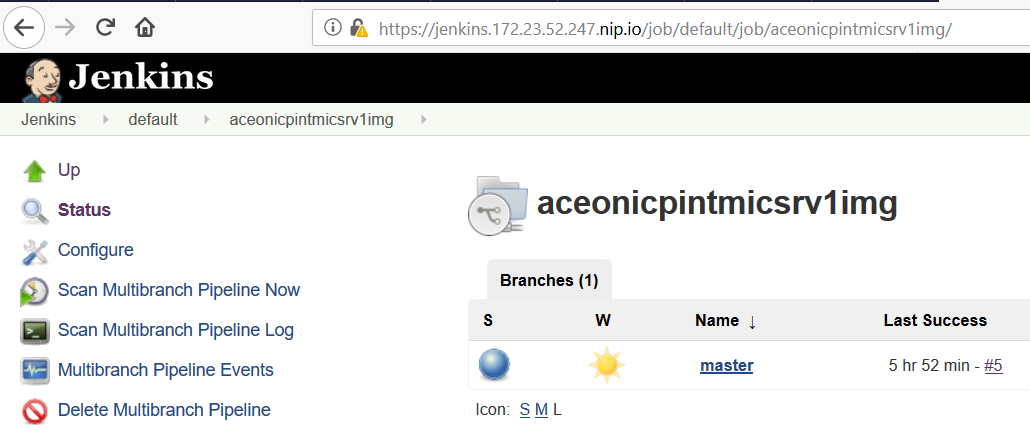
You need to copy the MicroService1.bar from the ACE Micro Service 1 source repository in the <https://github.com/DAVEXACOM/ACEonICPIntMicSrv1Img/tree/master/acesoe/binary> directory

### Microclimate Micro Service 1 Project





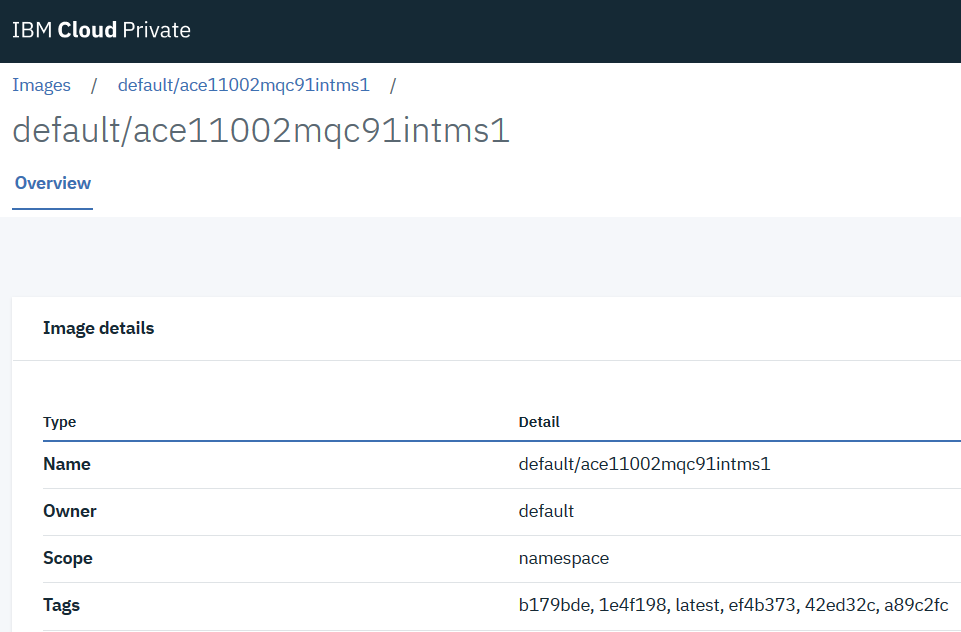
### Jenkins Pipeline for Micro Service 1





Note: The pipeline will try and deploy but fail. We don’t want this pipeline doing the deploy. This is an image build only

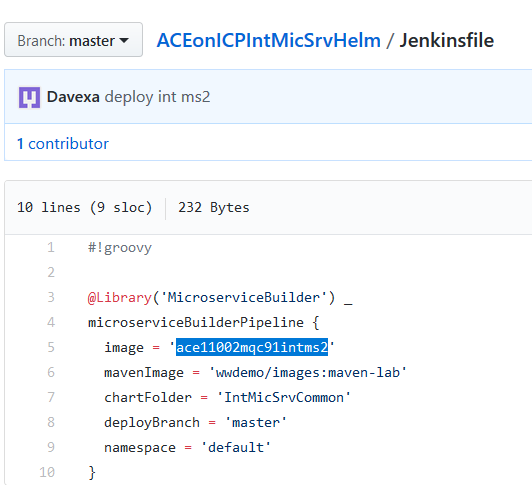
### Micro Service 1 Image on ICP



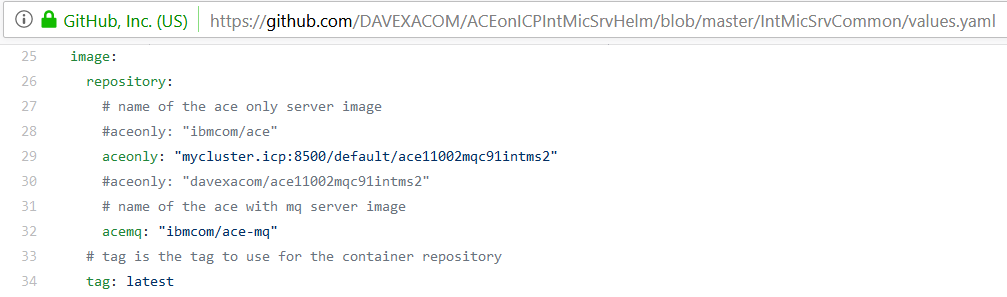
### Deploying Micro Service 1 on it’s own for testing – Use github Repos ACEonICPIntMicSrvHelm

<https://github.com/DAVEXACOM/ACEonICPIntMicSrvHelm>

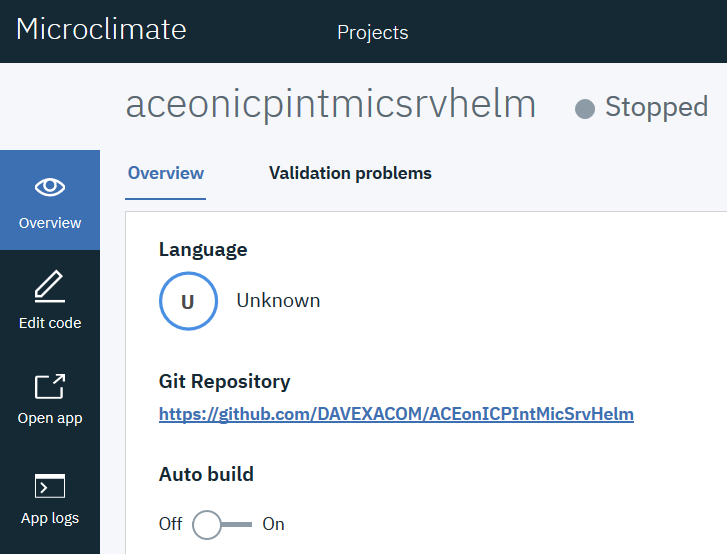
Use this Helm Chart Repos to toggle between Micro Service 1 and Micro Service 2 helm release deployments for testing.

Modify the Jenkins File (in this example you’ll need to switch from 2 to 1)

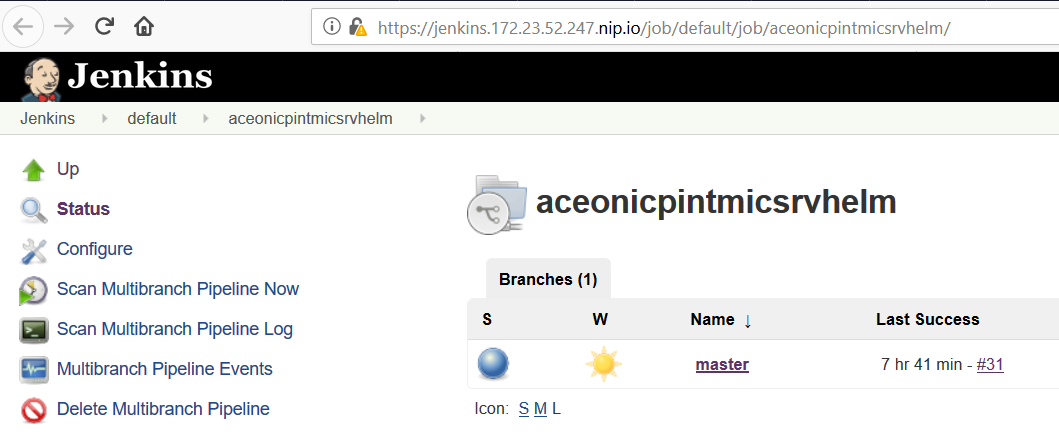
Modify the Values.yaml file

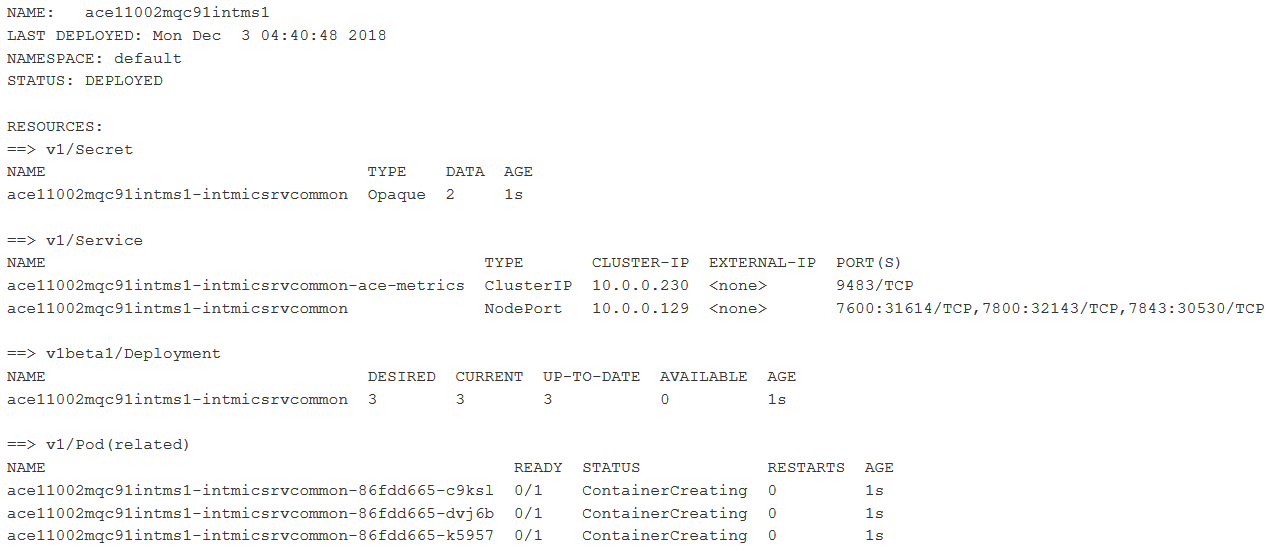


### Microclimate Project



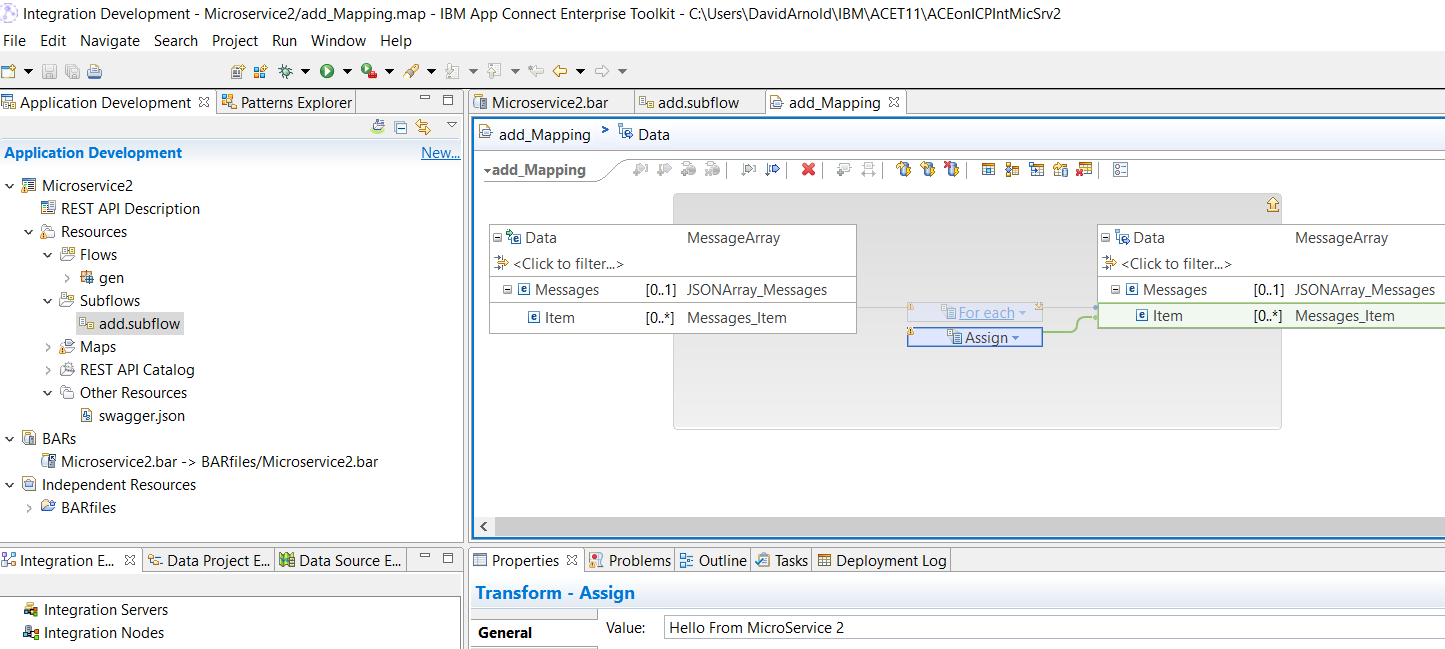
### Jenkins Pipeline





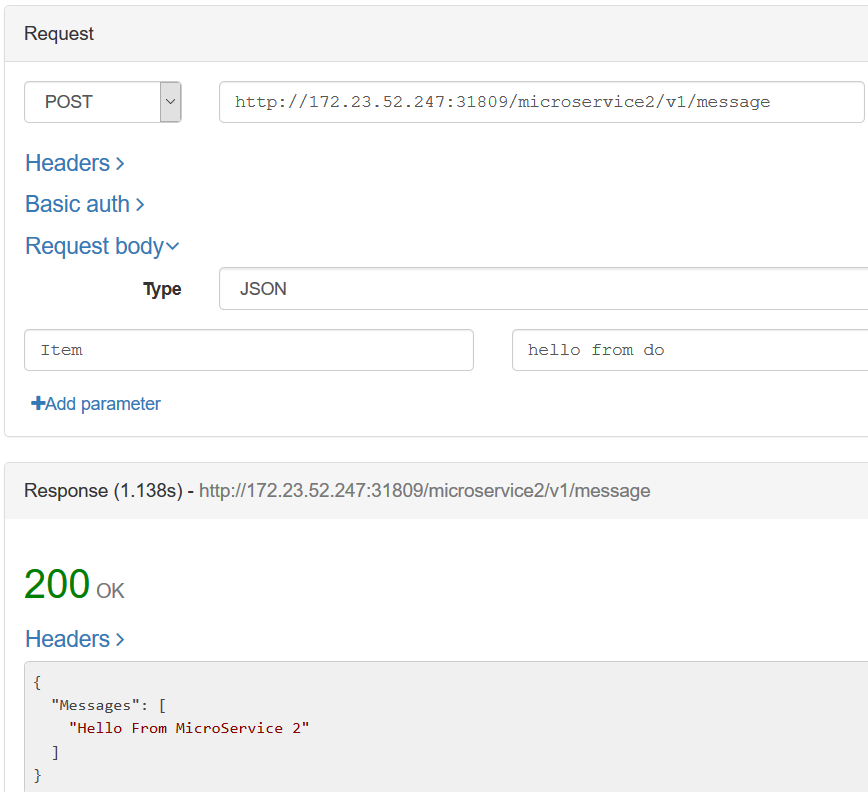
## ACE Integration Micro Service 2

Integration Micro Service 2 can be called directly as a rest service or access by calling Micro Service 1 (which in turn calls it)



### Testing Micro Service 2

Note it returns hello from Micro Service 2



### Github Source Repos for ACE Micro Service 2

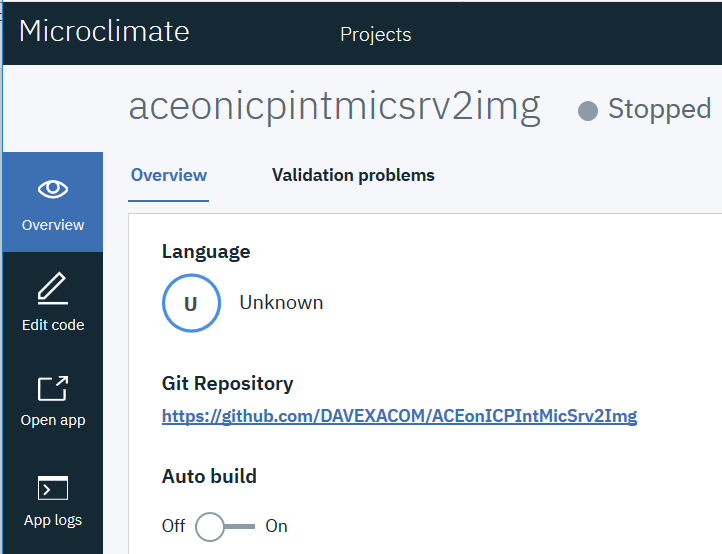
https://github.com/DAVEXACOM/ACEonICPIntMicSrv2

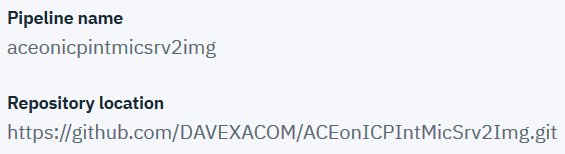
### Building the Image for Micro Service 2 – GitHub ACE Micro Service 1 Build

<https://github.com/DAVEXACOM/ACEonICPIntMicSrv2Img>

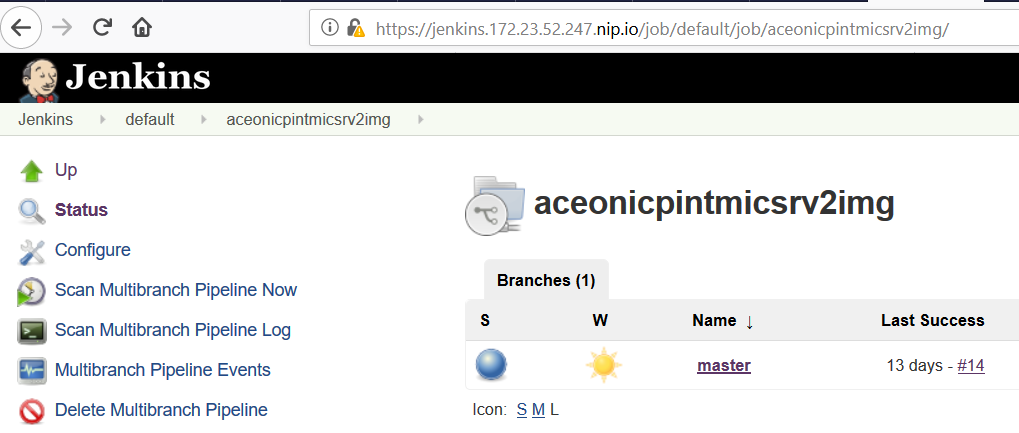
You need to copy the MicroService2.bar from the ACE Micro Service 2 source repository into the <https://github.com/DAVEXACOM/ACEonICPIntMicSrv2Img/tree/master/acesoe/binary> directory

### Microclimate Micro Service 2 Project





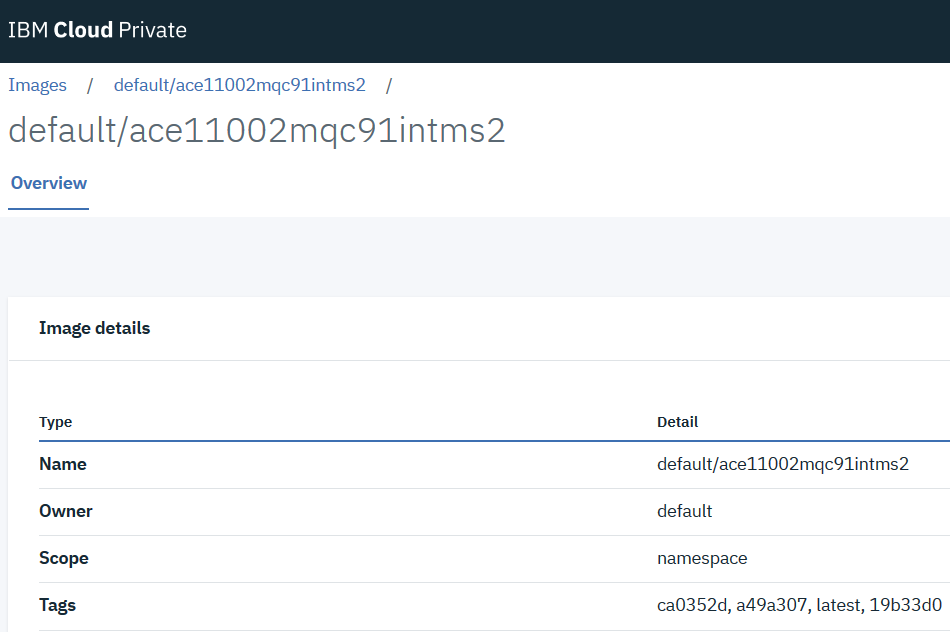
### Jenkins Pipeline for Micro Service 2





Note: The pipeline will try and deploy but fail. We don’t want this pipeline doing the deploy. This is an image build only

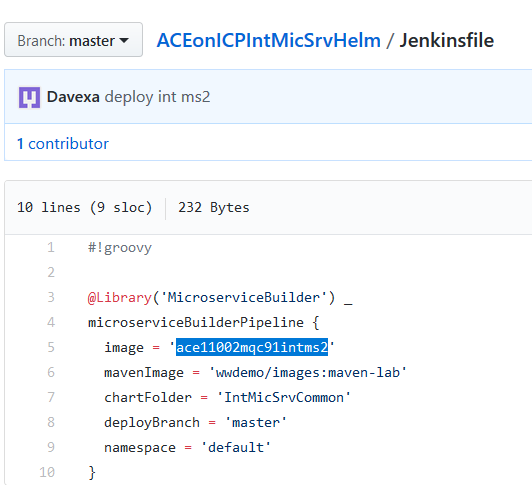
### Micro Service 2 Image on ICP



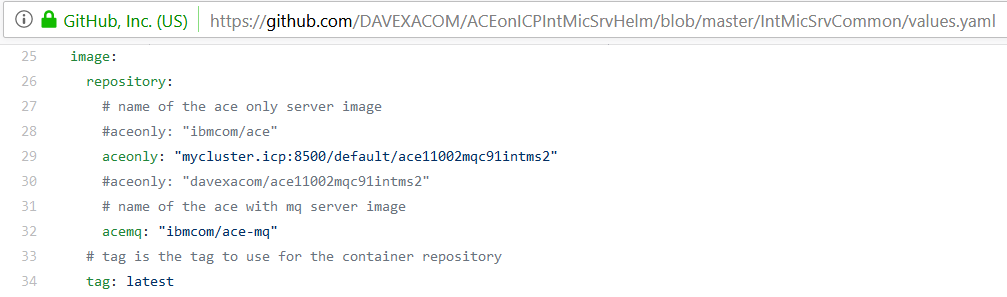
### Deploying Micro Service 2 on it’s own for testing – Use github Repos ACEonICPIntMicSrvHelm

<https://github.com/DAVEXACOM/ACEonICPIntMicSrvHelm>

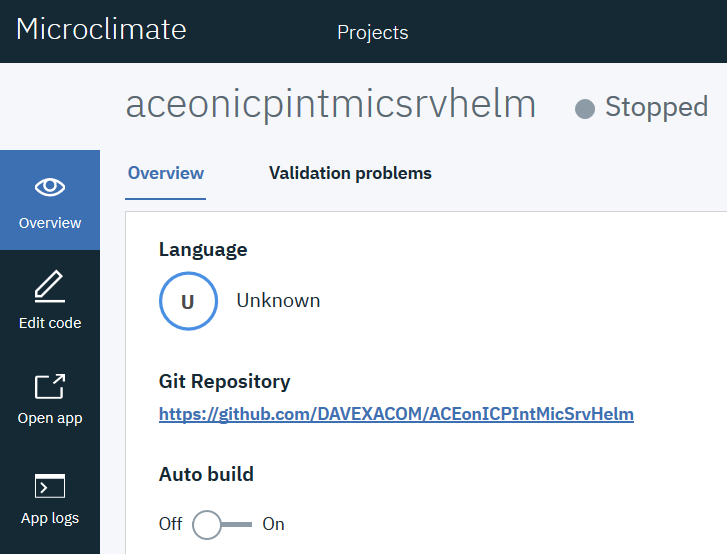
Use this Helm Chart Repos to toggle between Micro Service 1 and Micro Service 2 helm release deployments for testing.

Modify the Jenkins File 

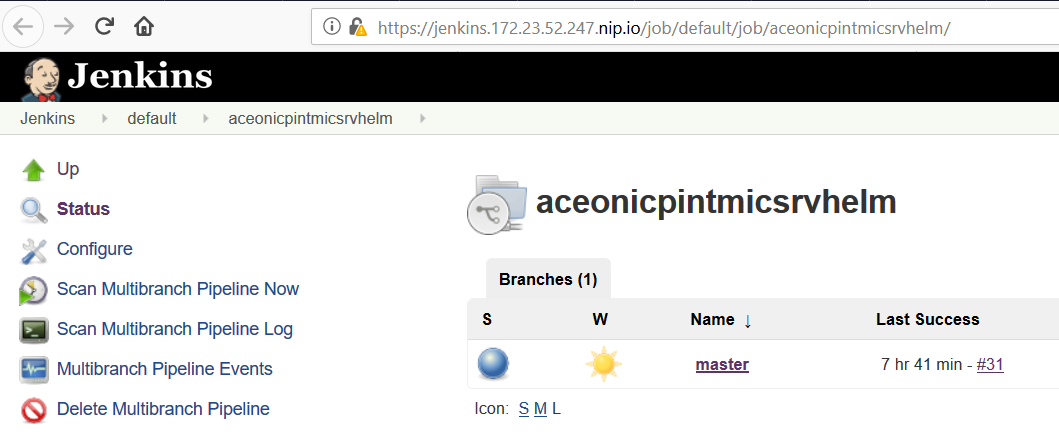
Modify the Values.yaml file

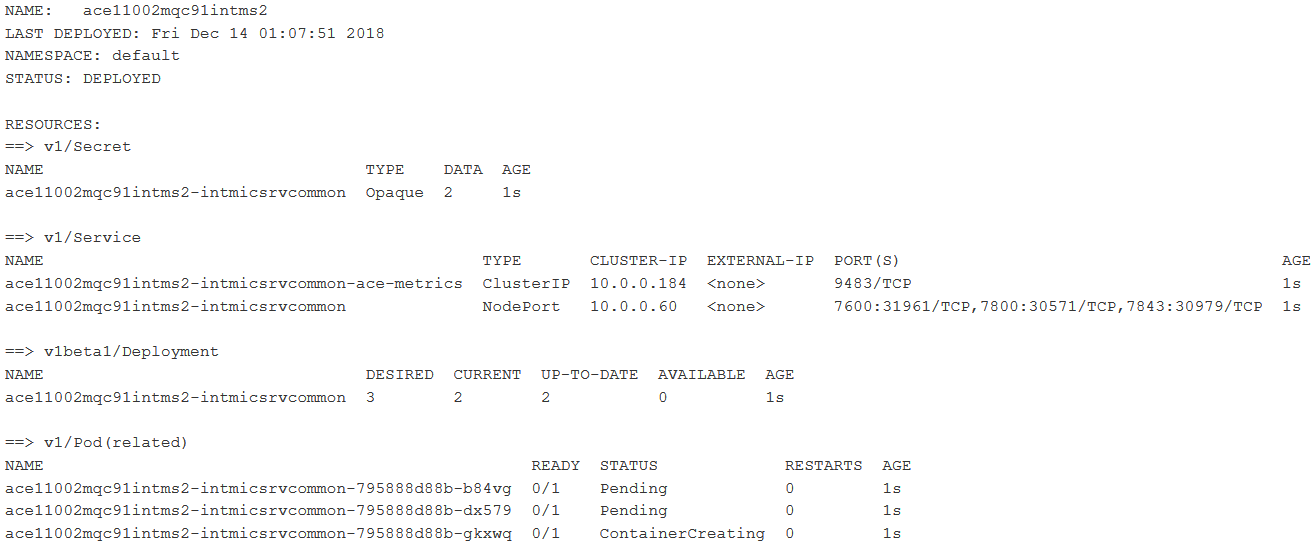


### Microclimate Project



### Jenkins Pipeline

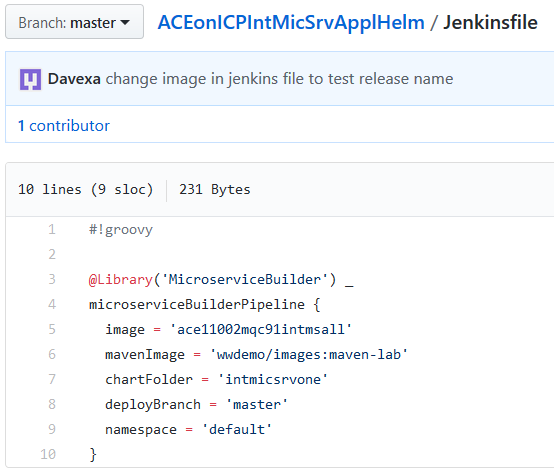




## Deploying Micro Service 1 and Micro Service 2 together as a Micro Services Appliction– Use github Repos ACEonICPIntMicSrvApplHelm

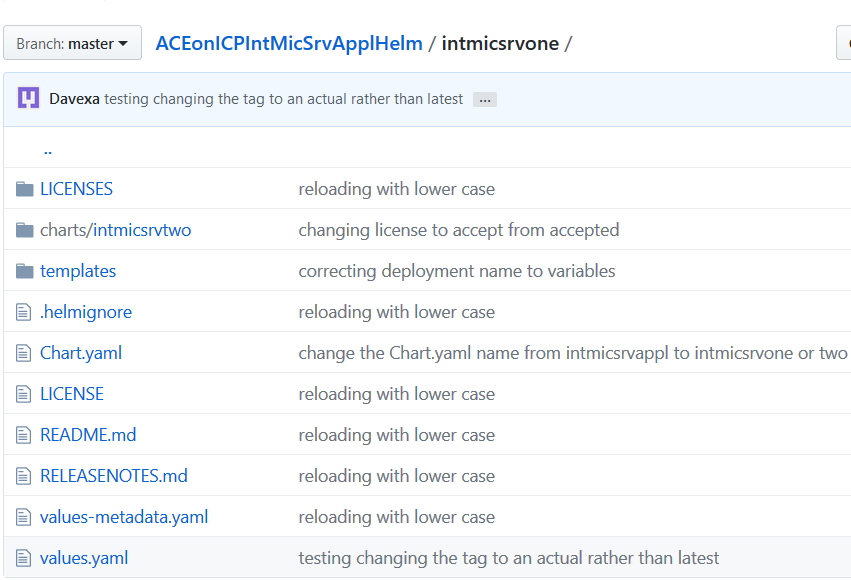
### Integration Micro Services Application Helm Charts GitHub Repository

<https://github.com/DAVEXACOM/ACEonICPIntMicSrvApplHelm>



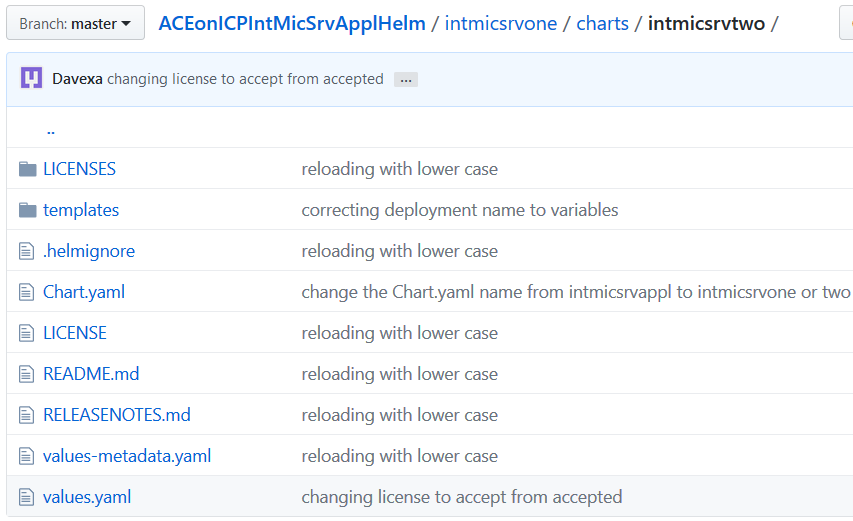
### Chart Files for Integration Micro Service 1

<https://github.com/DAVEXACOM/ACEonICPIntMicSrvApplHelm/tree/master/intmicsrvone>



### Sub Chart files for Integration Micro Service 2

<https://github.com/DAVEXACOM/ACEonICPIntMicSrvApplHelm/tree/master/intmicsrvone/charts/intmicsrvtwo>



### YAML Chart file modifications - How it hangs together

The following YAMl files have been modified from the OT4i content. The main modifications are in the templates YAMLs. Because the sub charts folder and content for integration service two is a copy of main charts files we need to ensure that the secrets and services etc are not created with the same names.

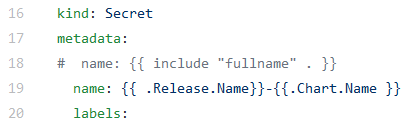
Chart – Chart.yaml



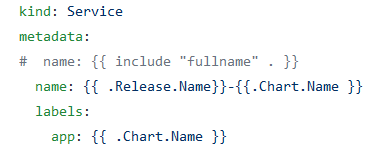
Sub chart – Chart.yaml



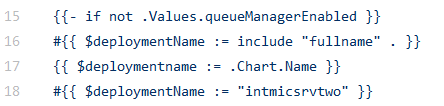
Secrets.yaml for example



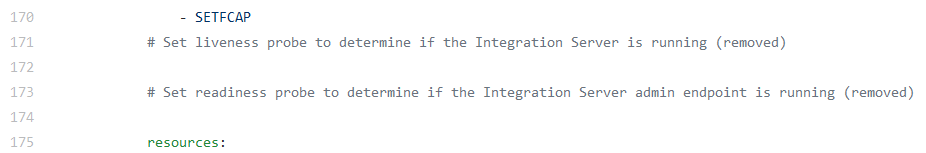
Service.yaml



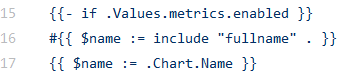
Deployment.yaml



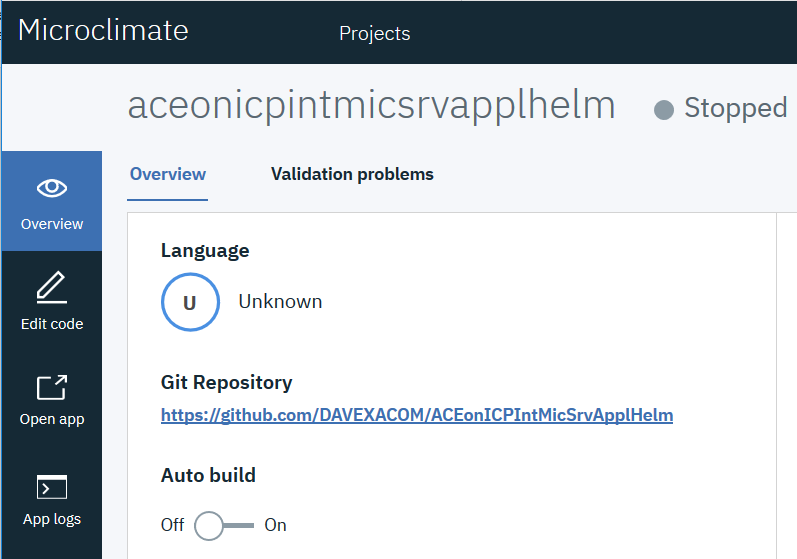
Also removed the cloud pack liveness and readiness probes we have our own in the SOE build image.

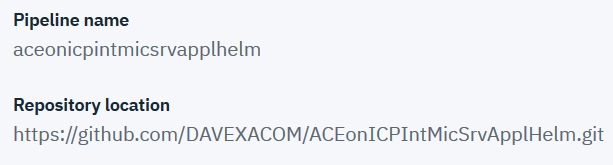


Metrics-service-ace.yaml

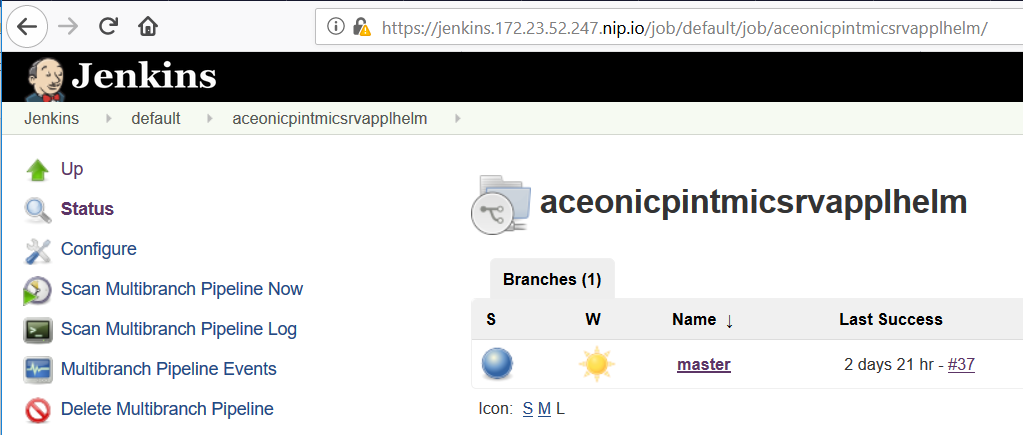


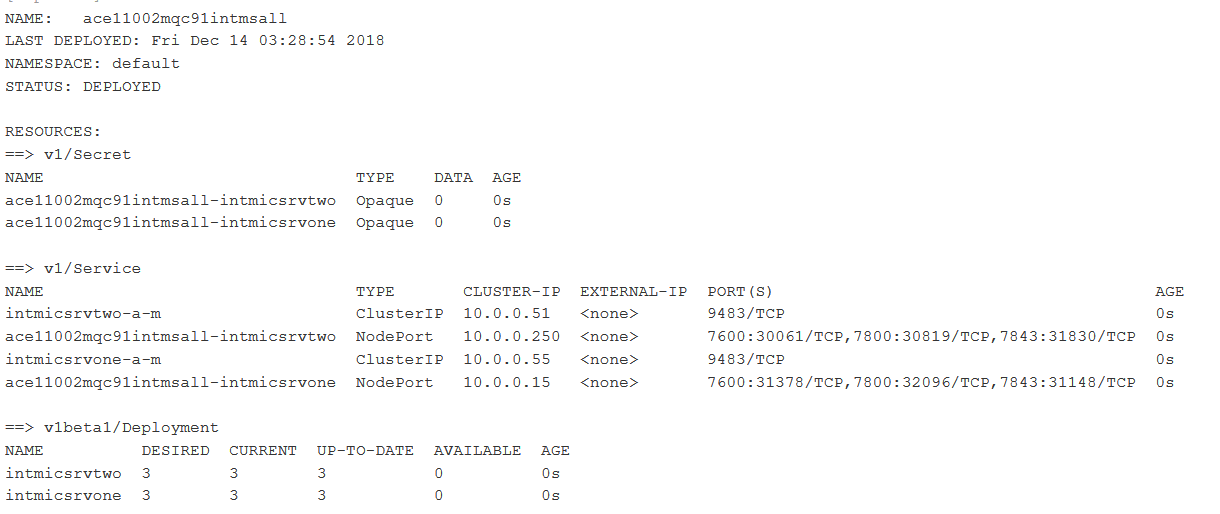
### Microclimate project aceonicpintmicsrvapplhelm



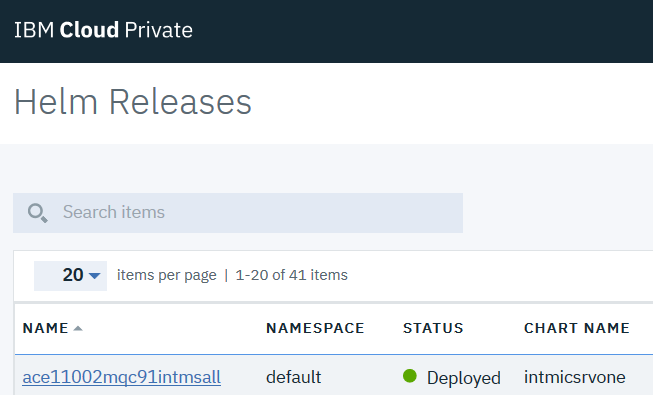


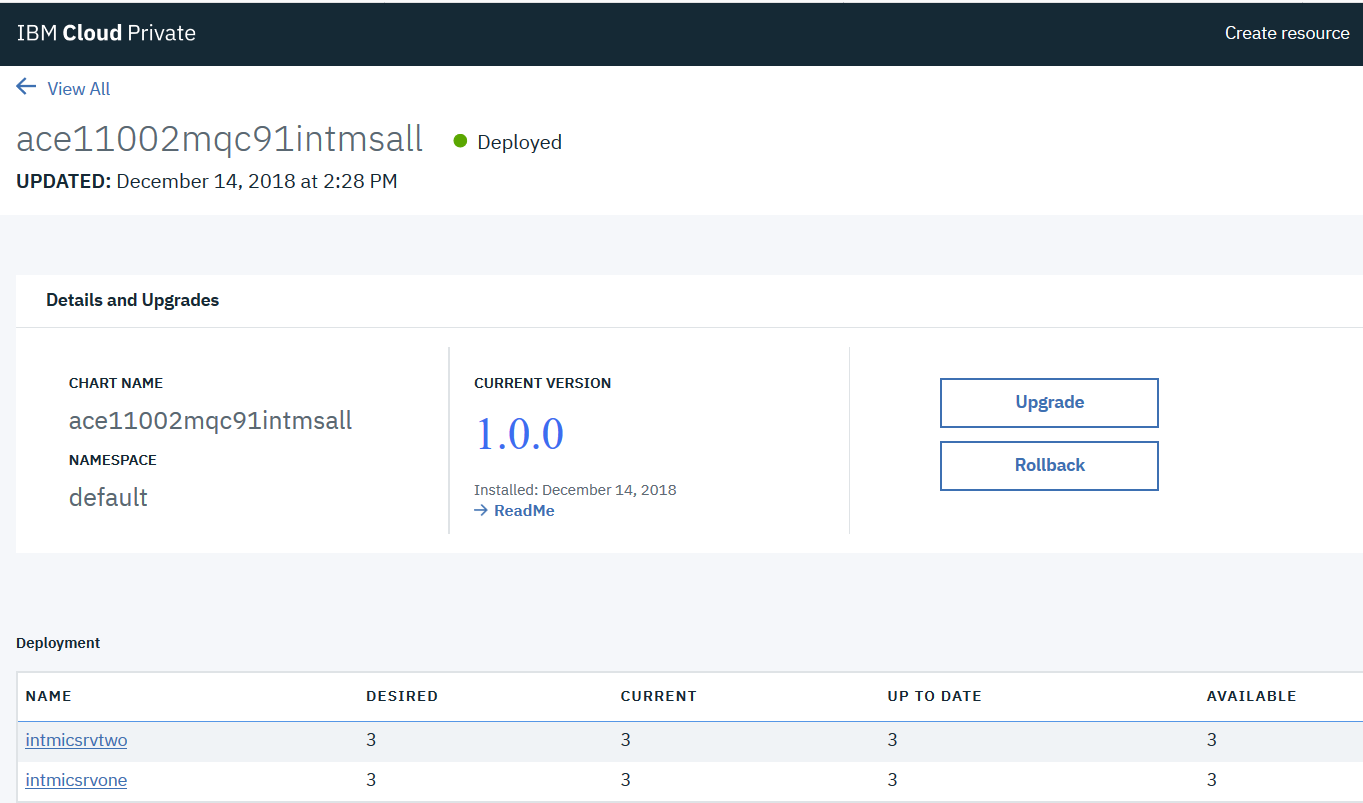
### Jenkins Pipeline - aceonicpintmicsrvapplhelm

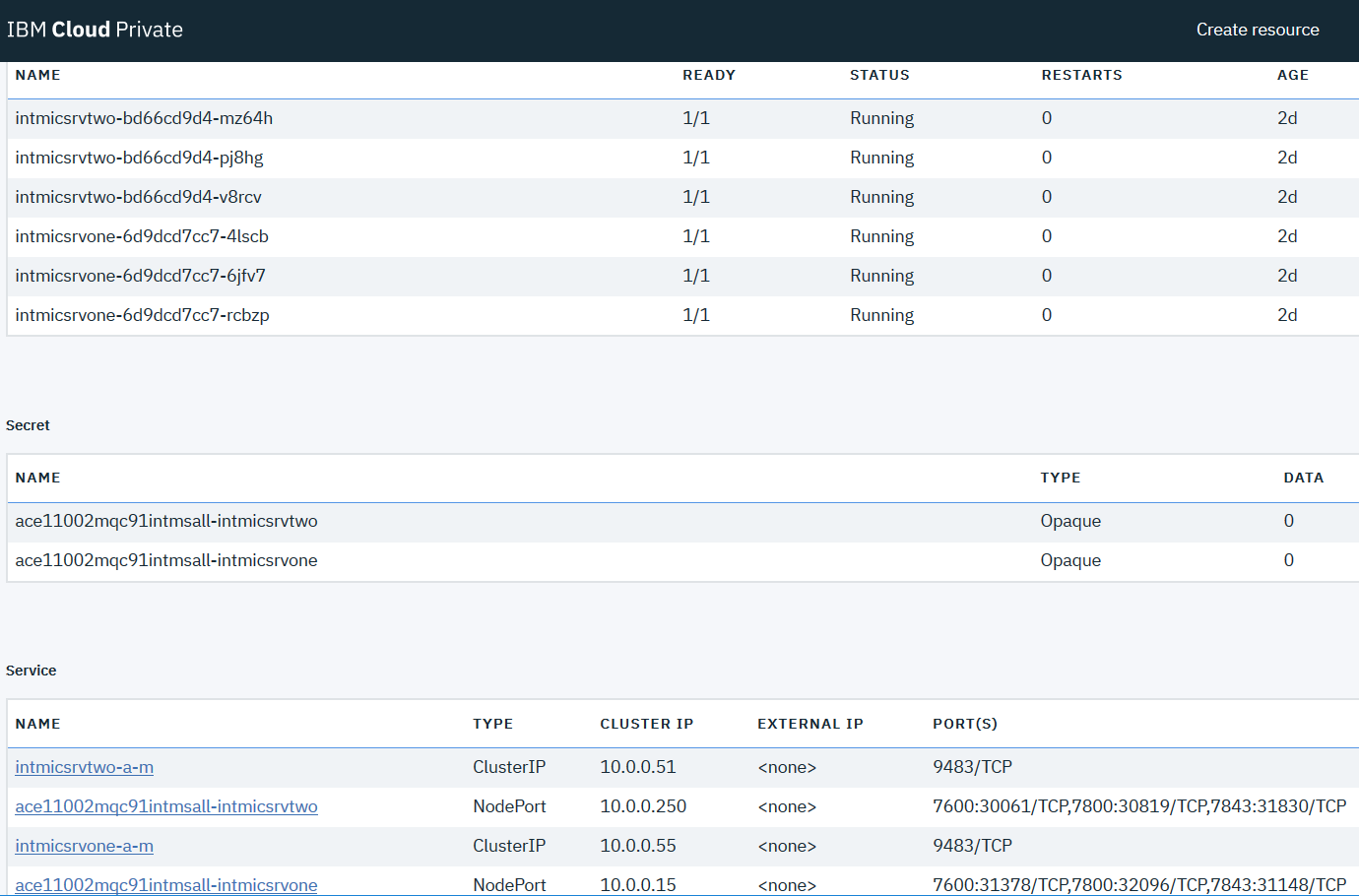




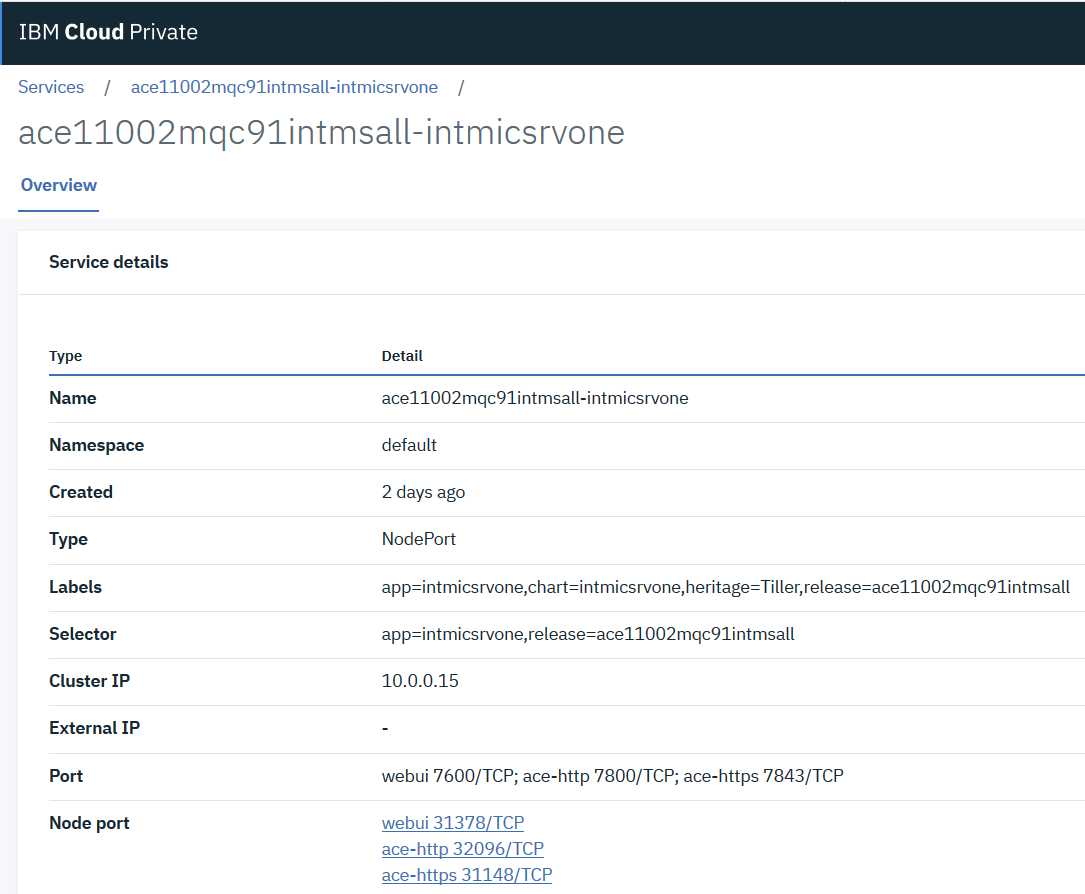
### Helm Release on ICP for Integration Micro Services Application

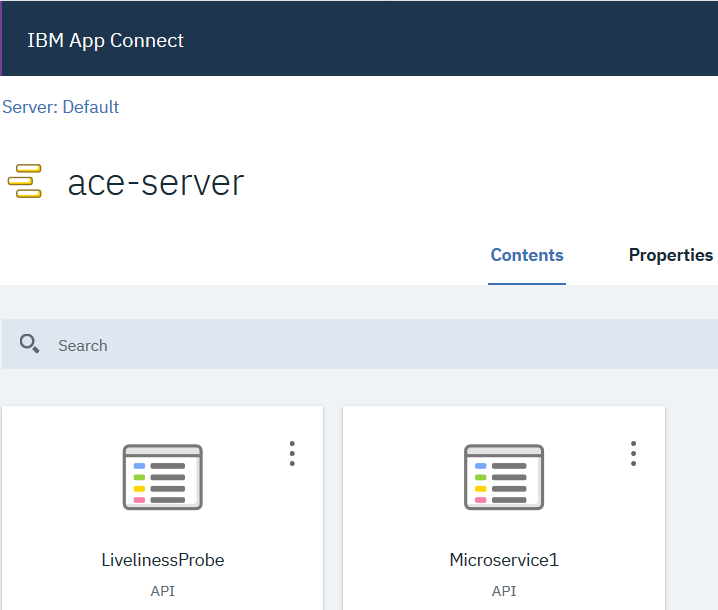




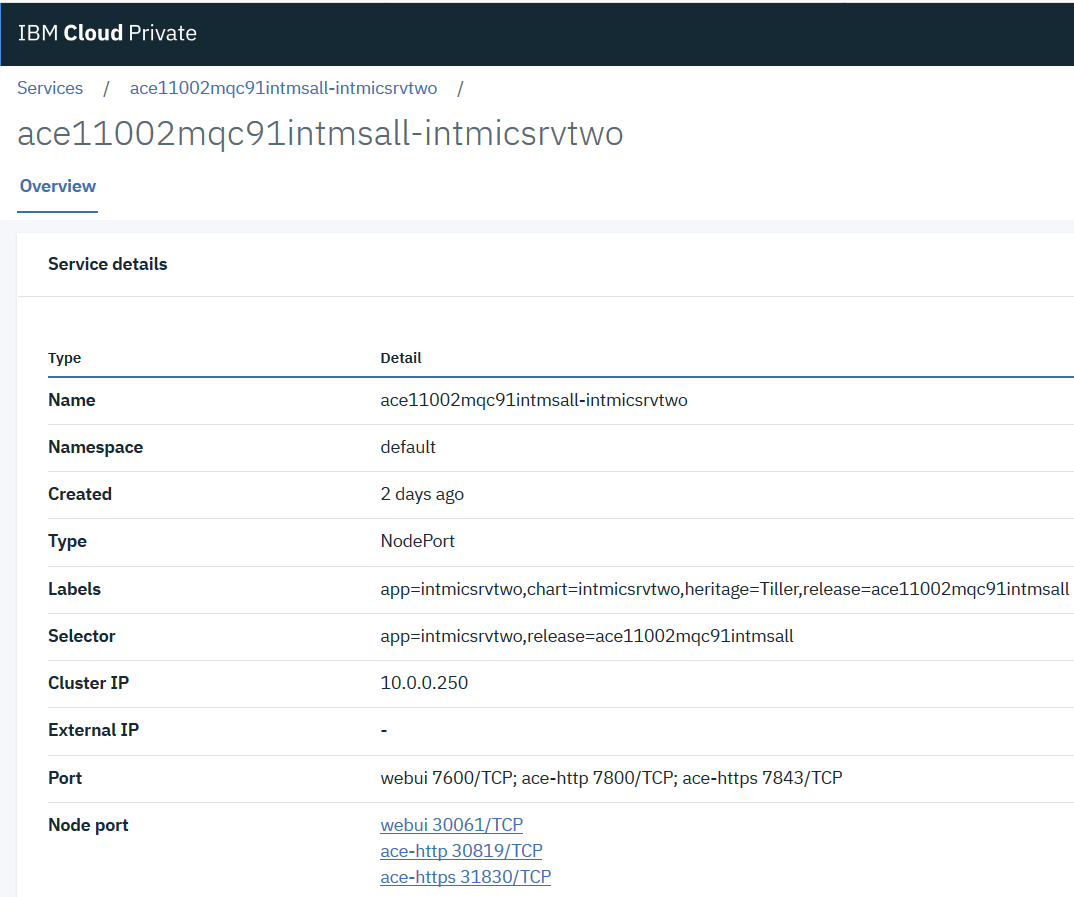


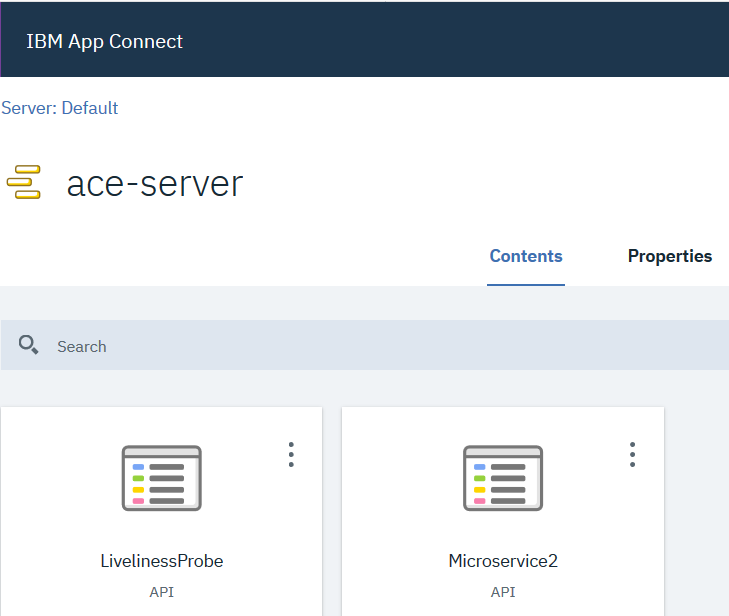
### Integration Micro Service One





### Integration Micro Service Two





### Testing the Integration Micro Services Application

