

Common_Cloud_Core

Table of contents

[Overview](#)

[Use Cases](#)

[Build Project](#)

[Define Agents](#)

[Define Pipeline](#)

[Define Stages](#)

[Run Build](#)

[Create Application](#)

[Create MicroService](#)

[Create Project](#)

[Debug Application](#)

[Debug MicroService](#)

[Deploy Application](#)

[Deploy MicroService](#)

[Modify Code](#)

[Publish Application](#)

[Publish MicroService](#)

[Test Application](#)

[Test MicroService](#)

[Actors](#)

[Developer](#)

[Solution](#)

Services

[Continuous Integration & Delivery](#)

[Registry](#)

Clouds

[Local Cloud](#)

[Dev Cloud](#)

[Test Cloud](#)

[Production Cloud](#)

Overview

Architectural Overview

Description

Users

- [Developer](#)

High level Use Cases

- [Create Application](#)
- [Create MicroService](#)
- [Create Project](#)
- [Debug Application](#)
- [Debug MicroService](#)
- [Deploy Application](#)
- [Deploy MicroService](#)
- [Modify Code](#)
- [Publish Application](#)
- [Publish MicroService](#)
- [Test Application](#)
- [Test MicroService](#)

Architecture: Cloud Aware Application Development Ecosystem

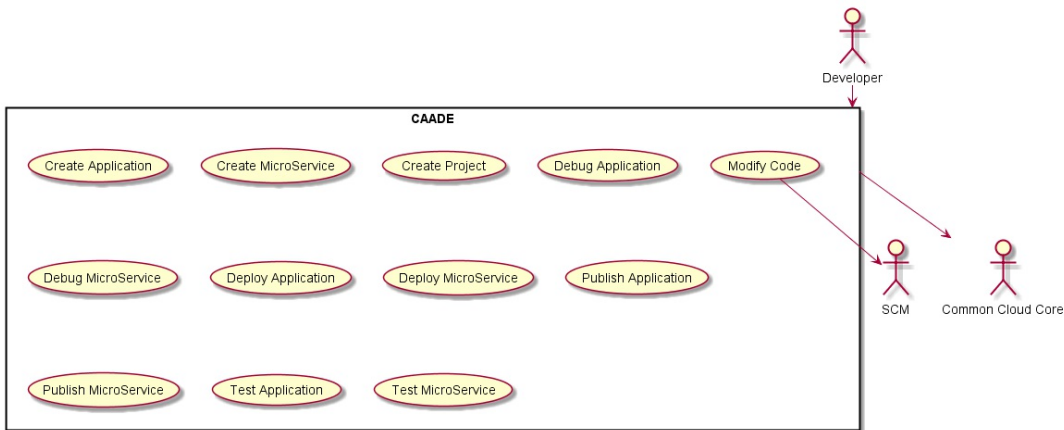


Figure: UseCases UseCases

Logical Architecture

Developers need to focus on the development of applications. When code is modified and checked into a code repository like github. A CI/CD system will automatically build, test and deploy the application, microservice or project. Multiple environments that have been created in the Common Cloud Core will be used by CAADE and the CI/CD to promote applications across the different environments.

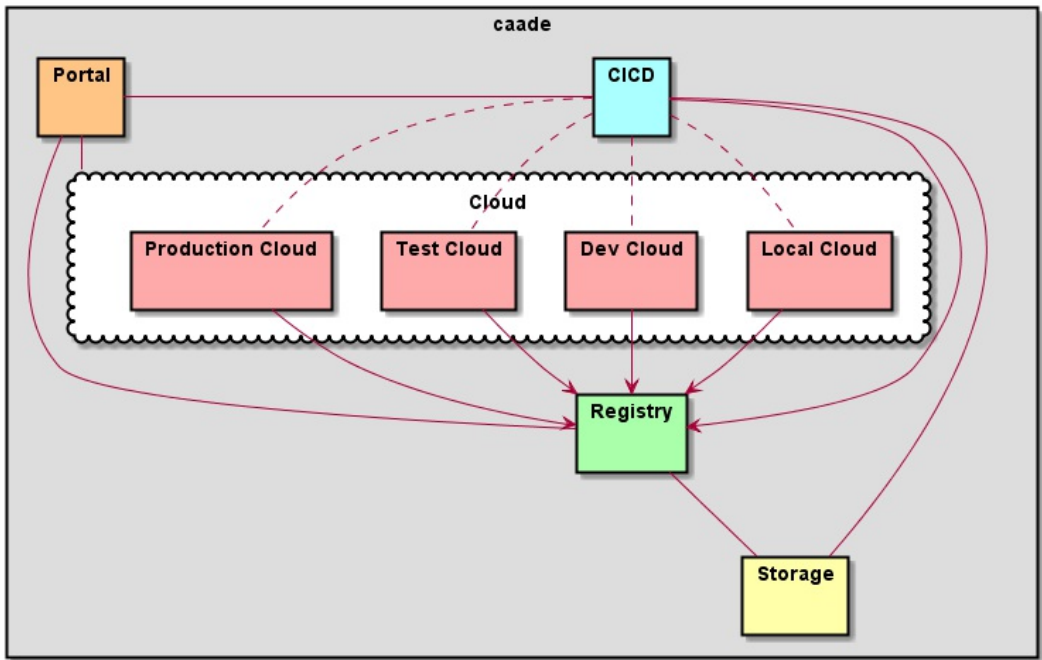


Figure: design Architecture

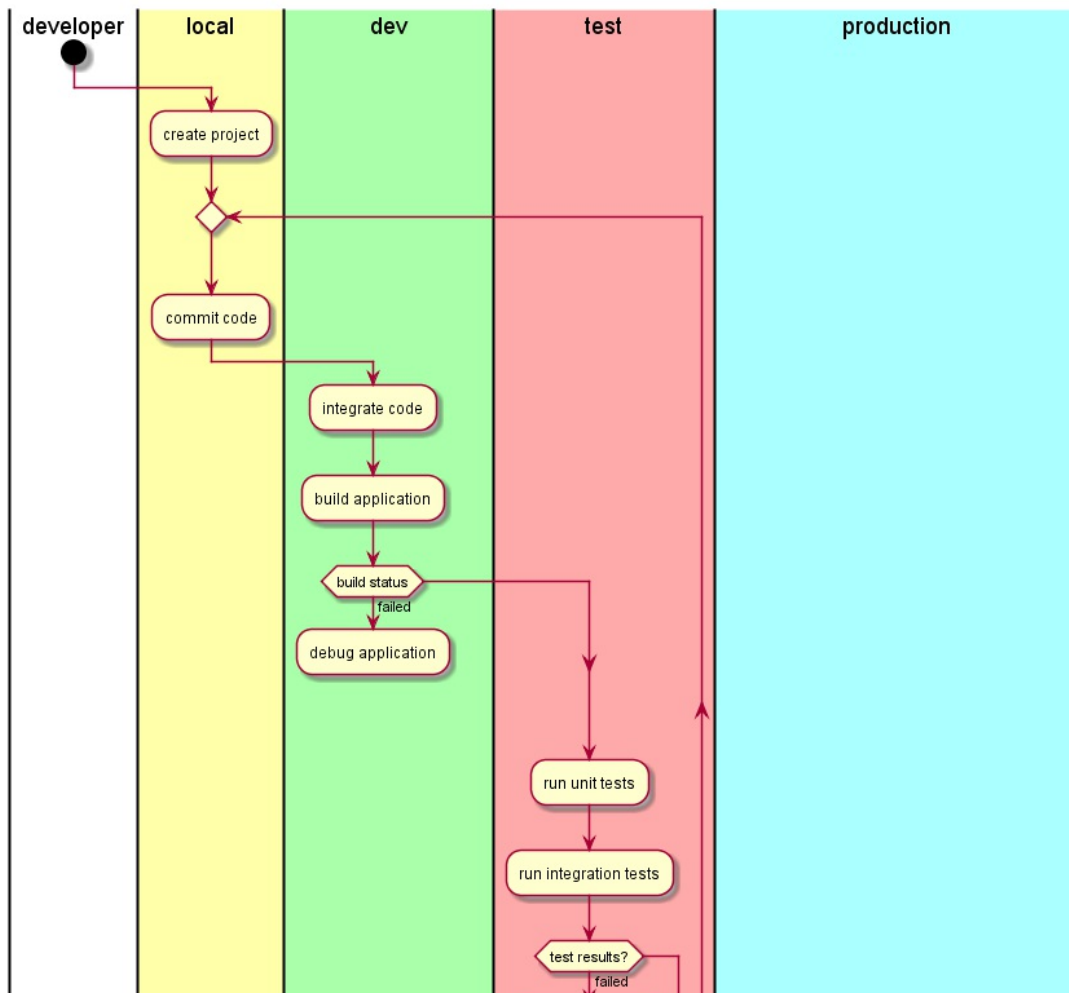
- [Common Cloud Core](#) - External

Architecture: Cloud Aware Application Development Ecosystem

- [CI/CD](#)
- [DevCloud](#)
- [LocalCloud](#)
- [ProductionCloud](#)
- [Registry](#)
- [SCM](#)
- [Test Cloud](#)

Process Architecture

This diagram shows how a developer interacts with CAADE to develop, test, and deploy cloud aware applications.



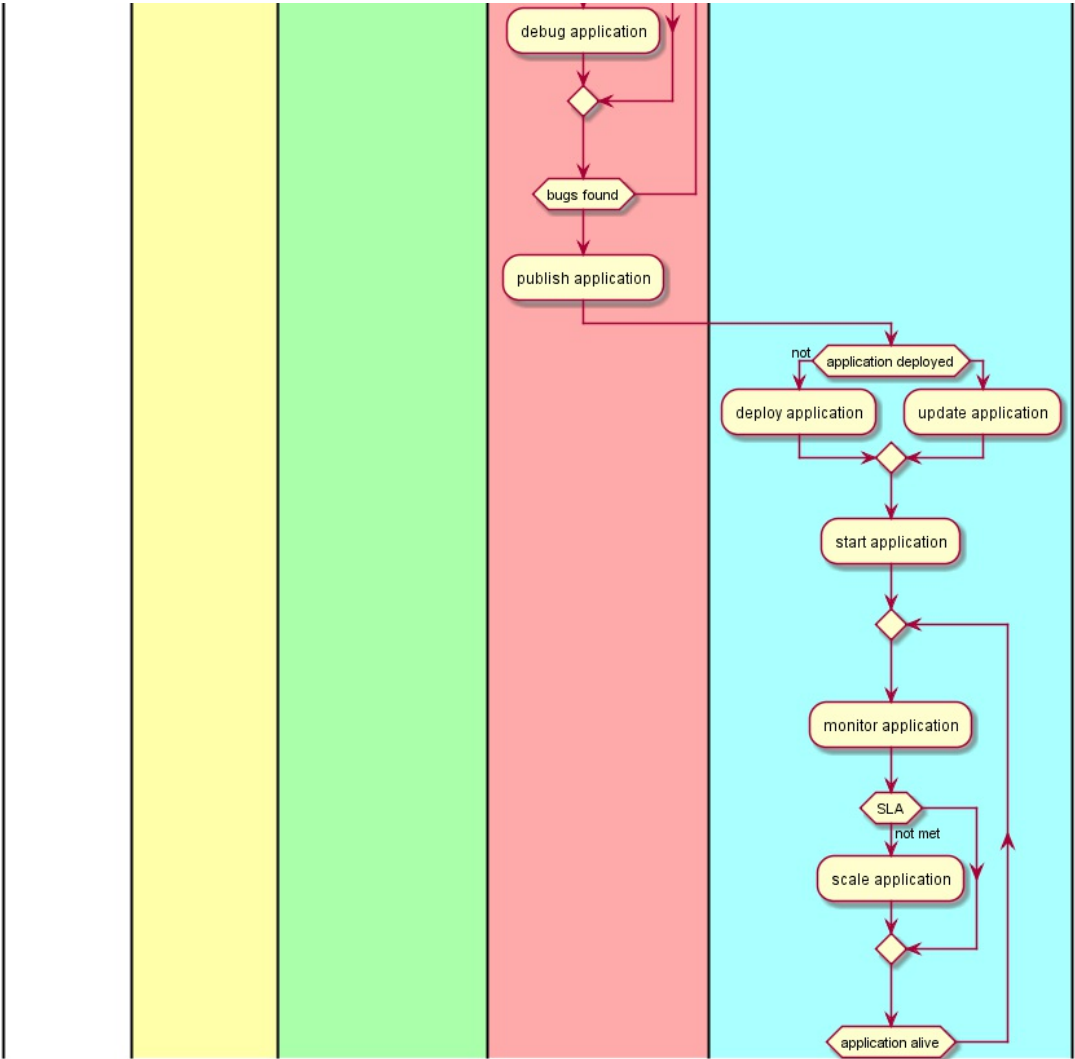


Figure: Solution Process

Deployment model

CAADE is made up a of a set of services and microservices to deliver capabilities to the Developer. The Service architect shown in the deployment model is an example of an implementation of a CAADE architecture.

Architecture: Cloud Aware Application Development Ecosystem

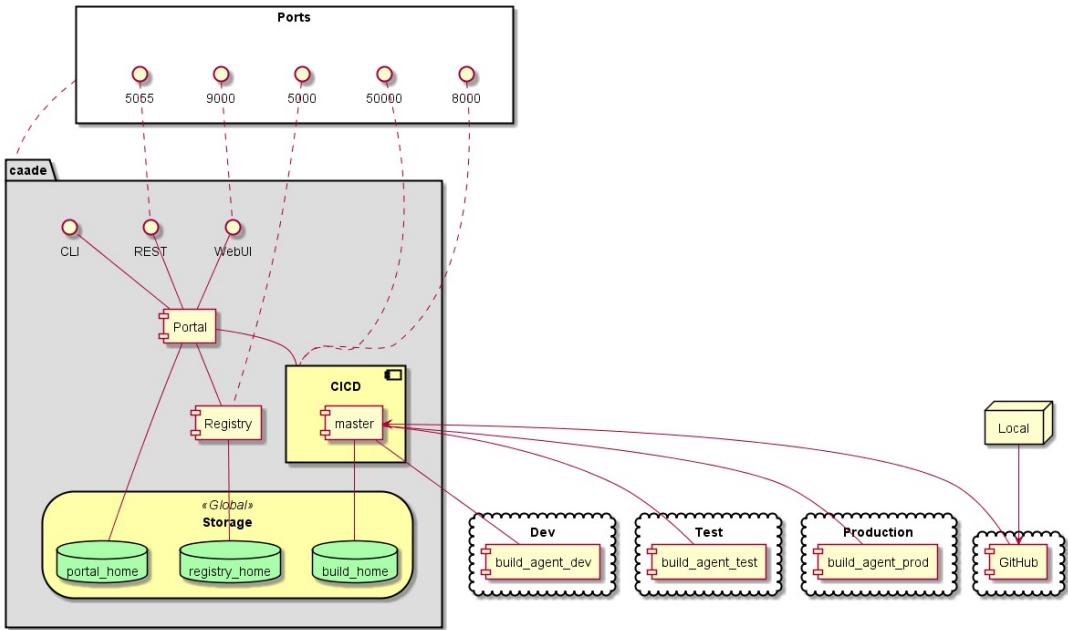


Figure: Solution Deployment

Physical Architecture

The physical architecture of CAAD is an example of a minimal hardware configuration that CAAD can be deployed.

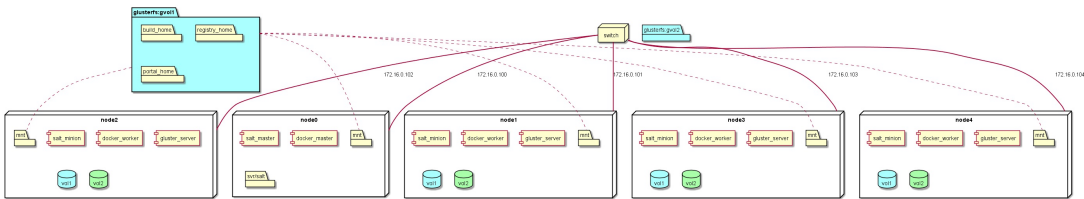


Figure: Solution Physical

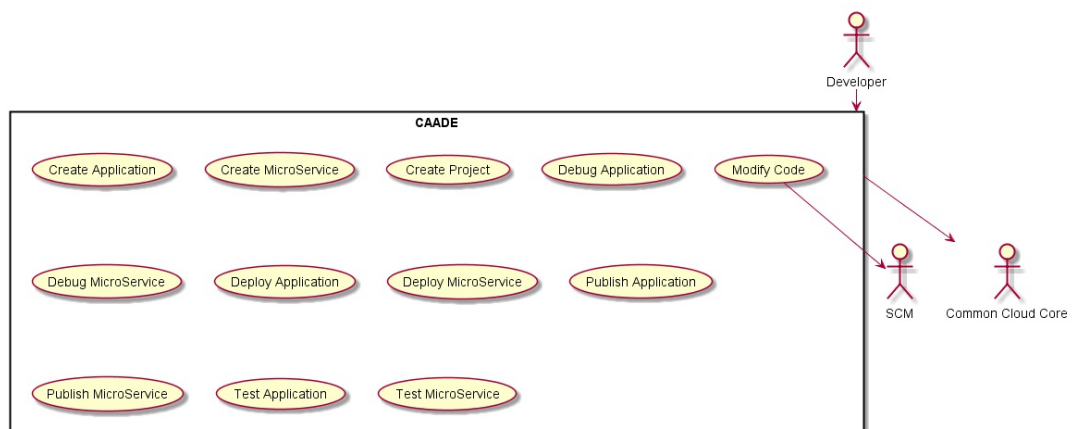
Use Cases

Use Cases for caade

High level use cases for the CAADE architecture.

Use Cases

- [Build Project](#)
- [Create Application](#)
- [Create MicroService](#)
- [Create Project](#)
- [Debug Application](#)
- [Debug MicroService](#)
- [Deploy Application](#)
- [Deploy MicroService](#)
- [Modify Code](#)
- [Publish Application](#)
- [Publish MicroService](#)
- [Test Application](#)
- [Test MicroService](#)



Architecture: Cloud Aware Application Development Ecosystem

Figure: UseCases UseCases

Build Project

Build-Project

Description

Actors

- Actors

Activities

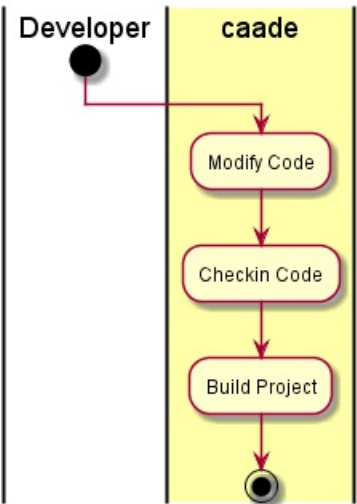


Figure: Build-Project Activities

- Activities

Detail Scenarios

- Scenarios

Systems Involved

- Systems

Define Agents

Scenario Define-Agents

Description

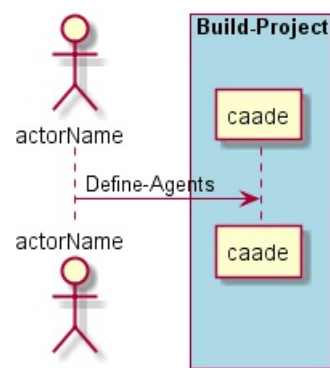


Figure: Build-Project Define-Agents

Define Pipeline

Scenario Define-Pipeline

Description

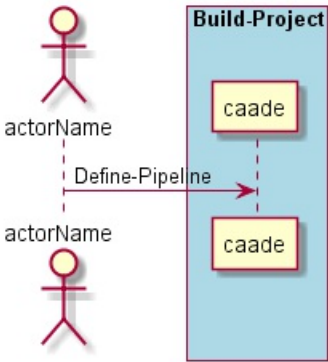


Figure: Build-Project Define-Pipeline

Define Stages

Scenario Define-Stages

Description

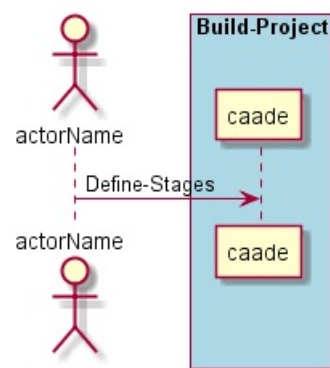


Figure: Build-Project Define-Stages

Run Build

Scenario Run-Build

Description

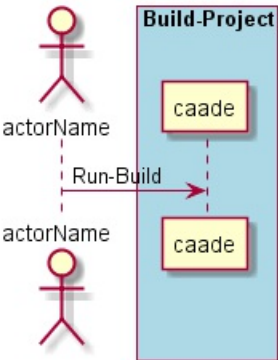


Figure: Build-Project Run-Build

Create Application

Create-Application

Description

Actors

- Actors

Activities

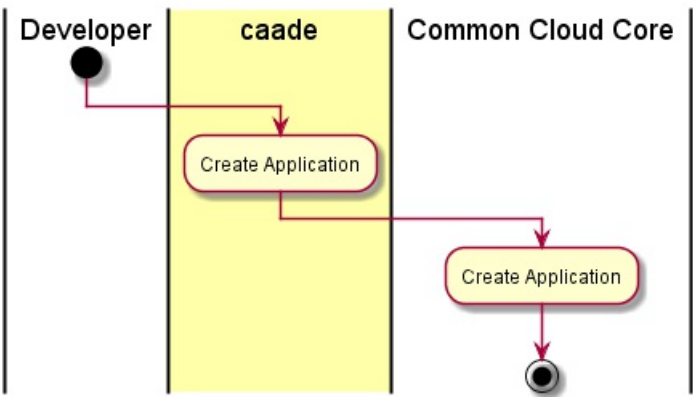


Figure: Create-Application Activities

- Activities

Detail Scenarios

- Scenarios

Systems Involved

- Systems

Create MicroService

Create-MicroService

Description

Actors

- *Actors*

Activities

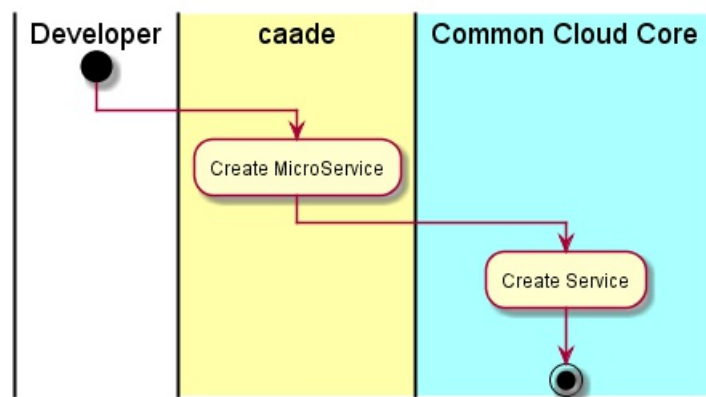


Figure: Create-MicroService Activities

- *Activities*

Detail Scenarios

- *Scenarios*

Systems Involved

- *Systems*

Create Project

Create-Project

Description

Actors

- Actors

Activities

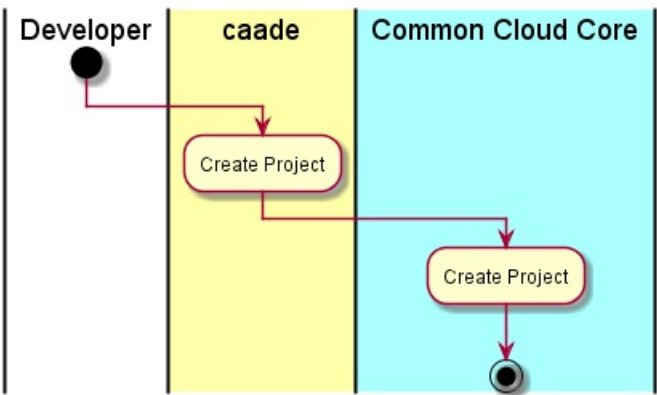


Figure: Create-Project Activities

- Activities

Detail Scenarios

- Scenarios

Systems Involved

- Systems

Debug Application

Debug-Application

Description

Actors

- Actors

Activities

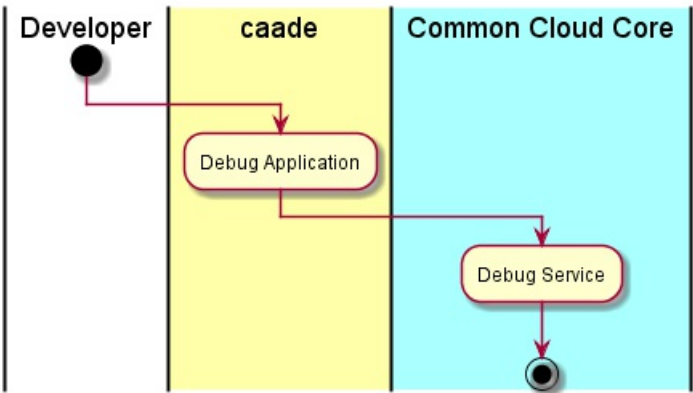


Figure: Debug-Application Activities

- Activities

Detail Scenarios

- Scenarios

Systems Involved

- Systems

Debug MicroService

Debug-MicroService

Description

Actors

- Actors

Activities

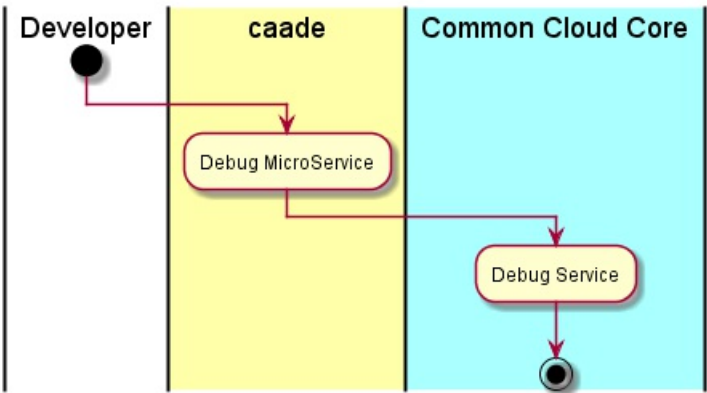


Figure: Debug-MicroService Activities

- Activities

Detail Scenarios

- Scenarios

Systems Involved

- Systems

Deploy Application

Deploy-Application

Description

Actors

- Actors

Activities

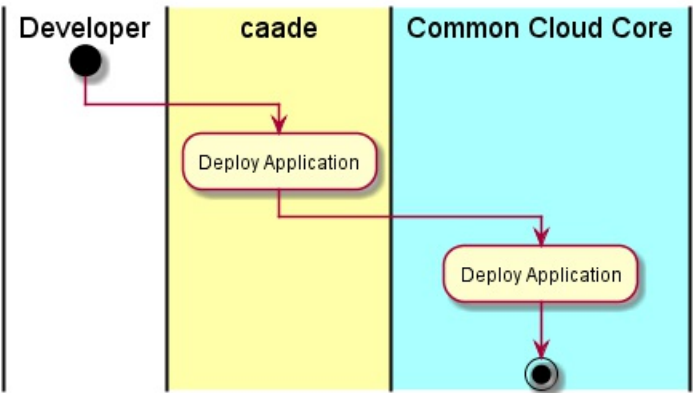


Figure: Deploy-Application Activities

- Activities

Detail Scenarios

- Scenarios

Systems Involved

- Systems

Deploy MicroService

Deploy-MicroService

Description

Actors

- Actors

Activities

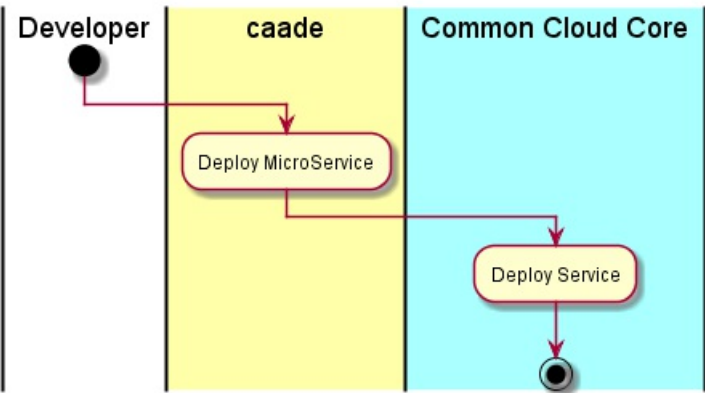


Figure: Deploy-MicroService Activities

- Activities

Detail Scenarios

- Scenarios

Systems Involved

- Systems

Modify Code

Modify-Code

Description

Actors

- Actors

Activities

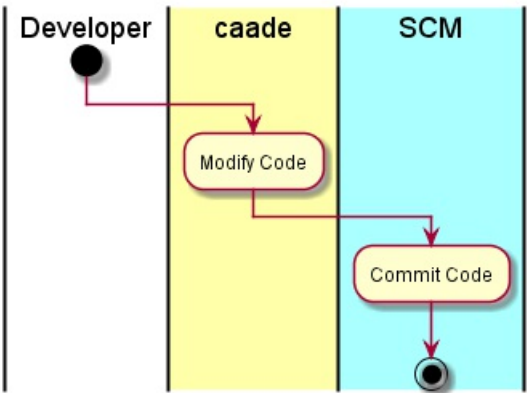


Figure: Modify-Code Activities

- Activities

Detail Scenarios

- Scenarios

Systems Involved

- Systems

Publish Application

Publish-Application

Description

Actors

- Actors

Activities

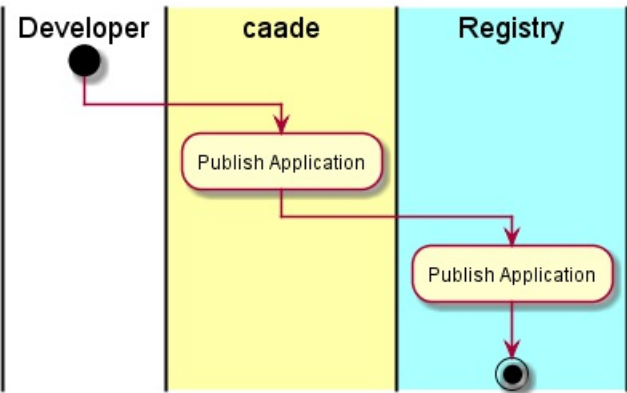


Figure: Publish-Application Activities

- Activities

Detail Scenarios

- Scenarios

Systems Involved

- Systems

Publish MicroService

Publish-MircoService

Description

Actors

- *Actors*

Activities

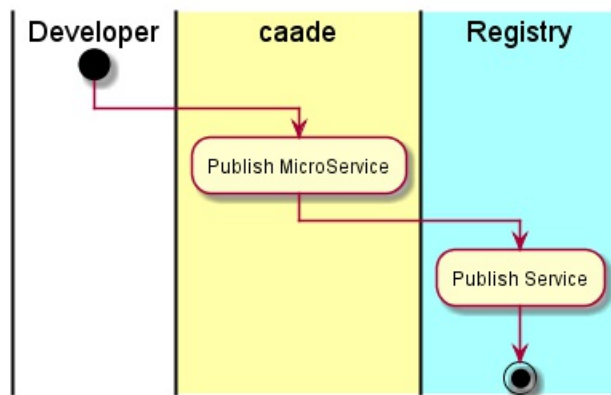


Figure: Publish-MicroService Activities

- *Activities*

Detail Scenarios

- *Scenarios*

Systems Involved

- *Systems*

Test Application

Test-Application

Description

Actors

- Actors

Activities

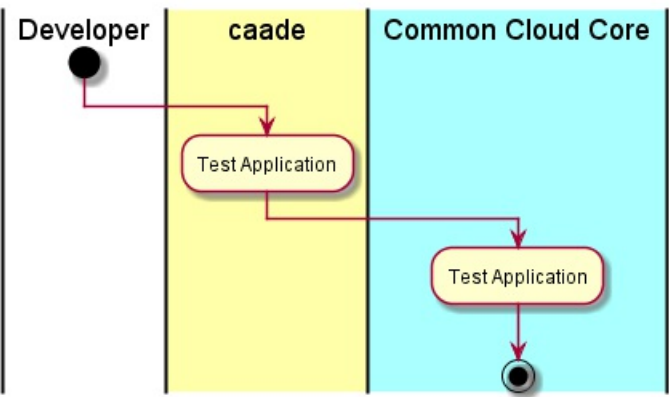


Figure: Test-Application Activities

- Activities

Detail Scenarios

- Scenarios

Systems Involved

- Systems

Test MicroService

Test-MicroService

Description

Actors

- Actors

Activities

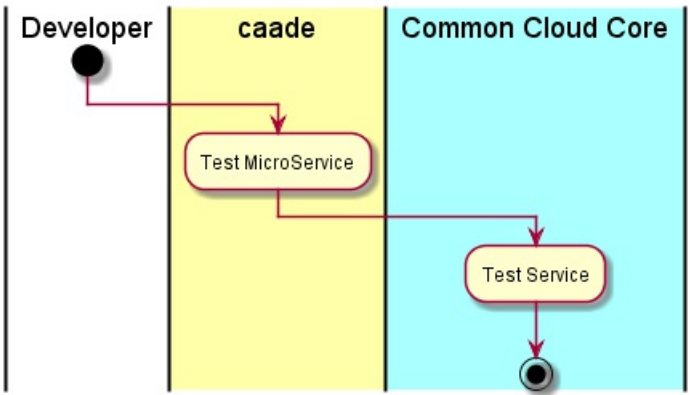


Figure: Test-MicroService Activities

- Activities

Detail Scenarios

- Scenarios

Systems Involved

- Systems

Actors

Actors

- [Developer](#)

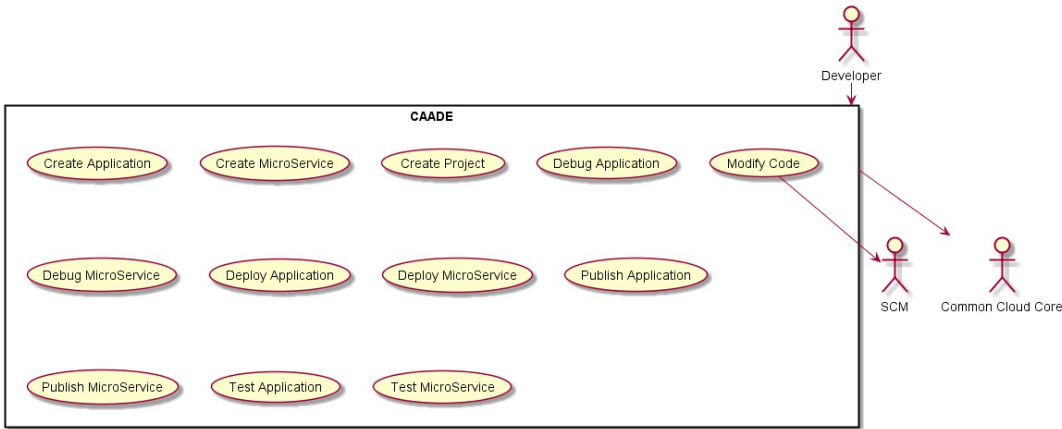


Figure: UseCases UseCases

Developer

Developer

Description

Use Cases

- UseCases

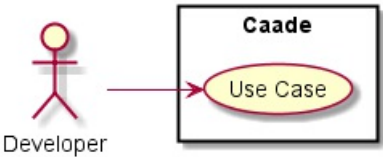


Figure: Developer UseCases

Activities

description

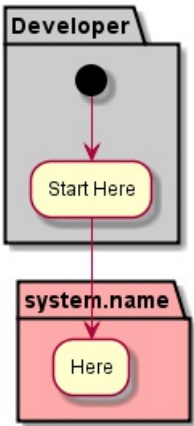


Figure: Developer Activity

Workflow

description

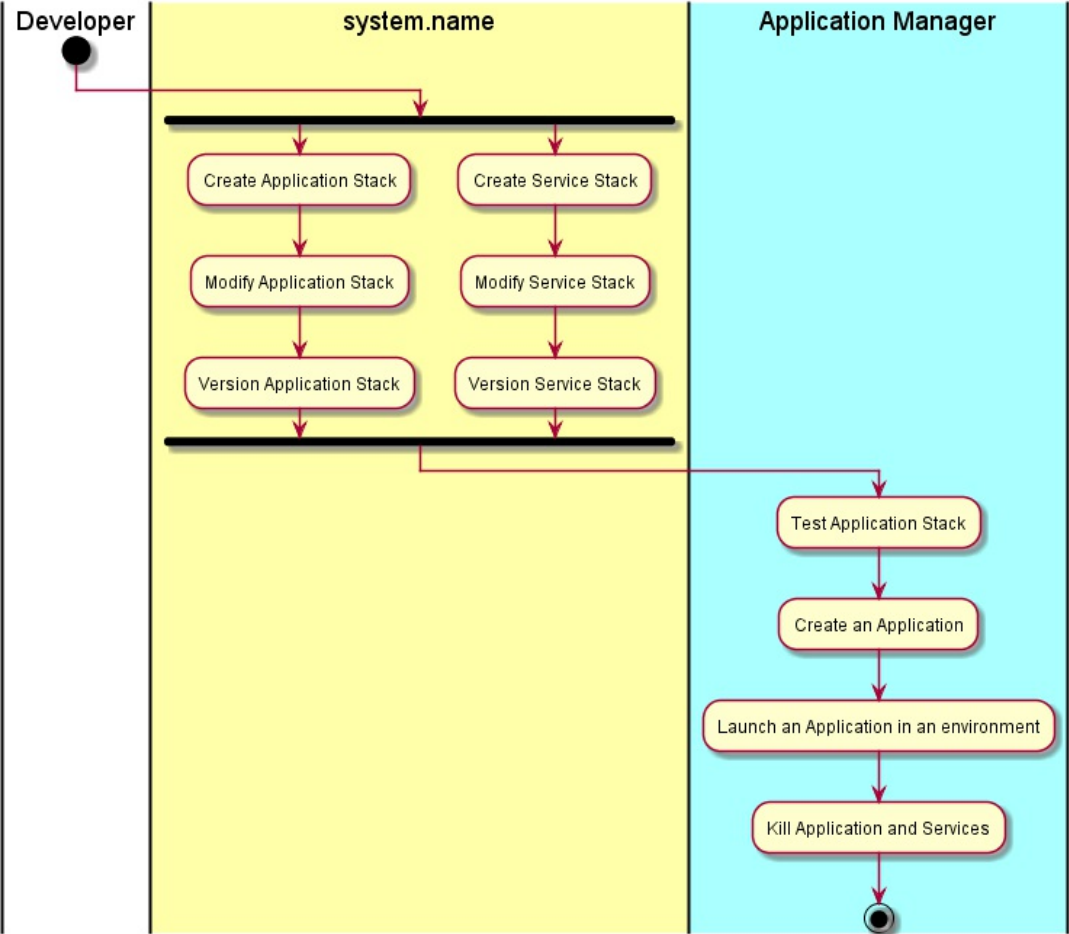


Figure: Developer Workflow

User Interface

TBD

Command Line Interface

TBD

Solution

Architectural Overview

Description

Users

- Actor

High level Use Cases

- UseCases

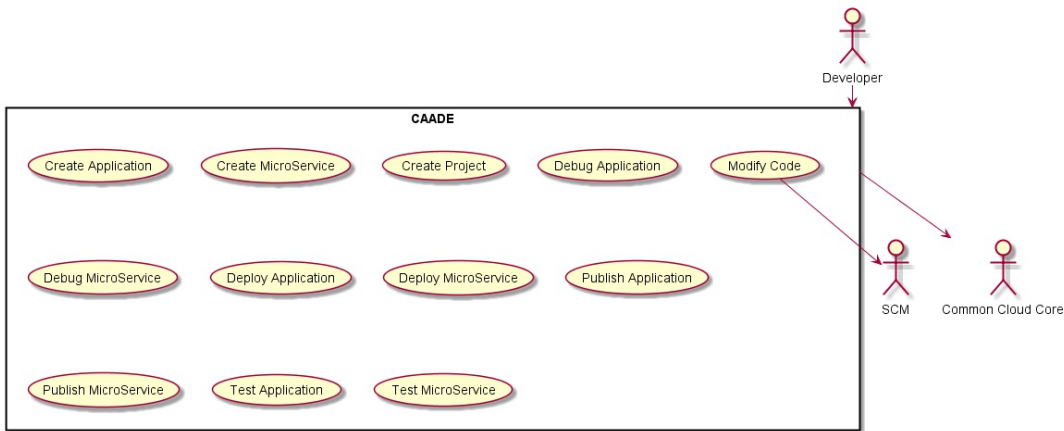


Figure: UseCases UseCases

Logical Architecture

Description

Architecture: Cloud Aware Application Development Ecosystem

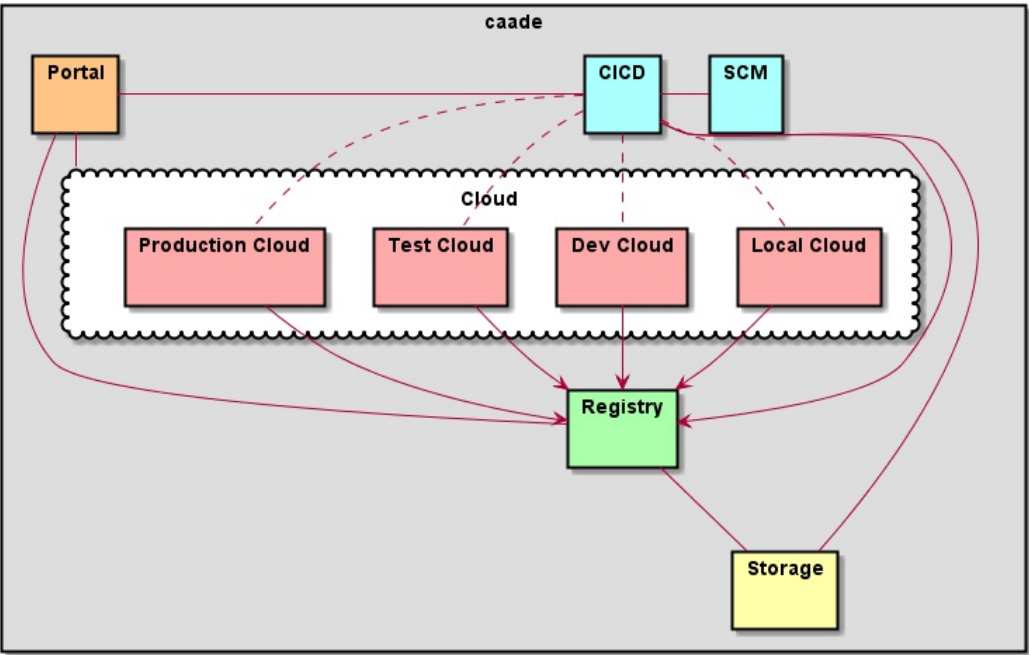
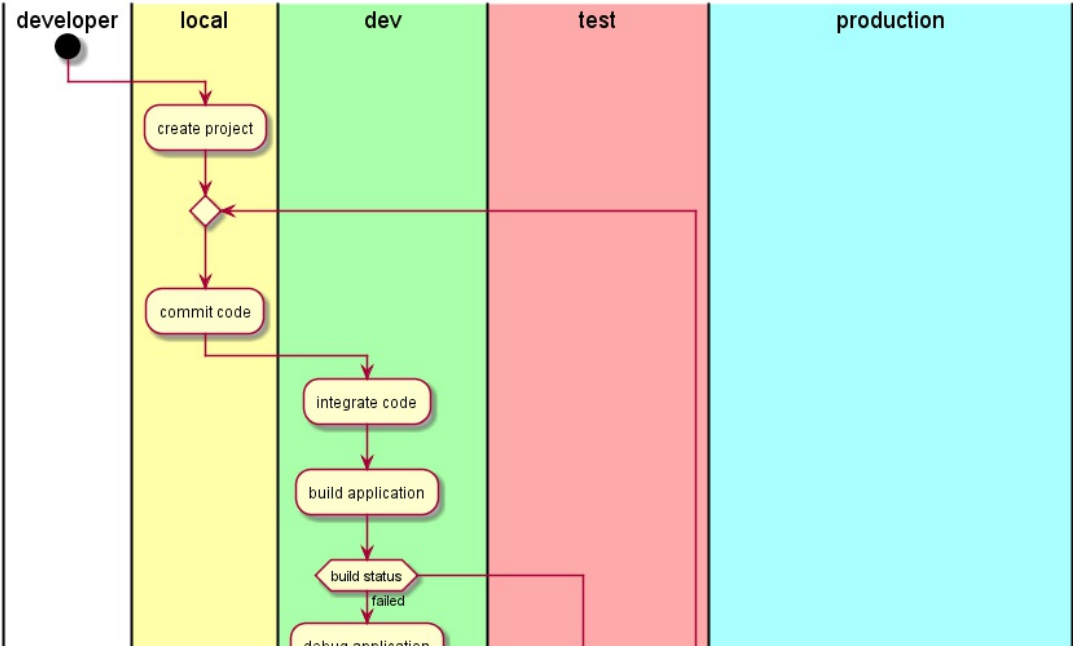


Figure: Solution Logical

Systems

Process Architecture



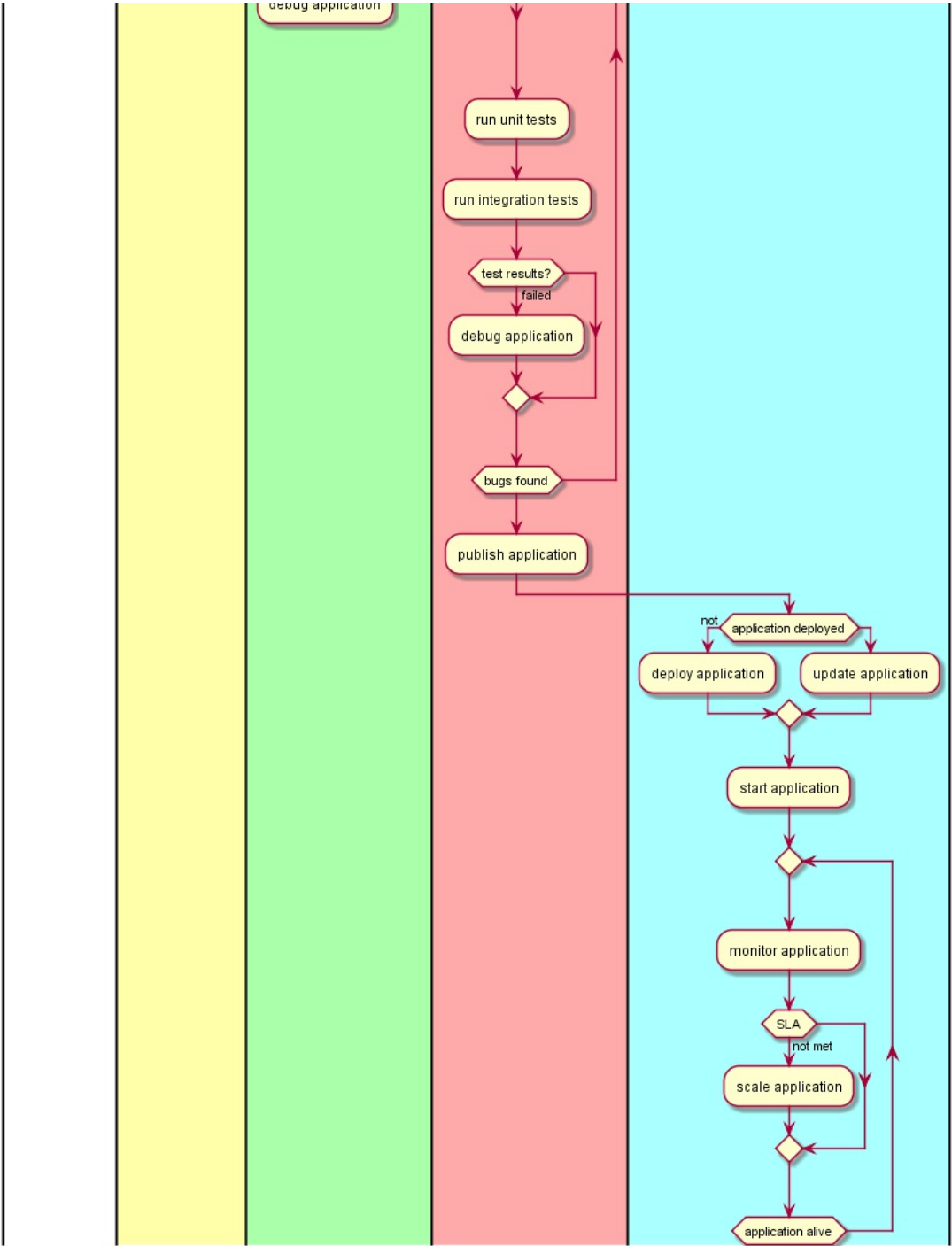


Figure: Solution Process

Deployment model

Architecture: Cloud Aware Application Development Ecosystem

Description

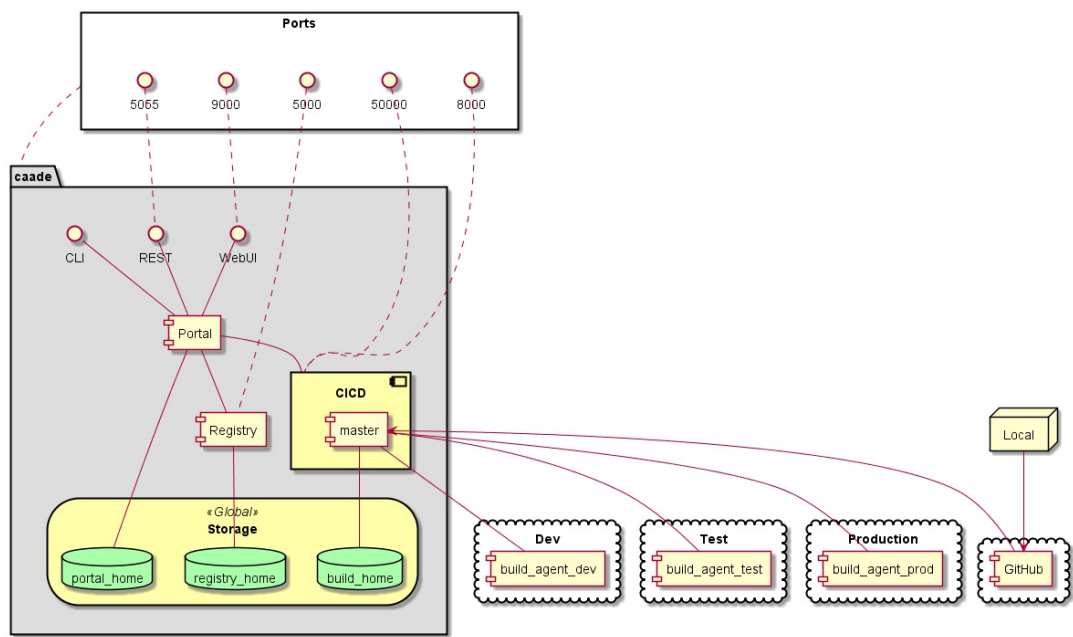


Figure: Solution Deployment

Physical Architecture

Description

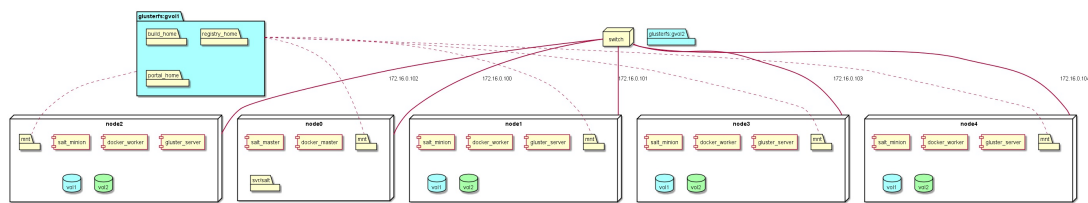


Figure: Solution Physical

Systems

Deployment_

This is a Reference Architecture for the CAADe solution using Salt, Docker, Jenkins, and Gluster

Salt Stack

Install Salt Master on Node 0

Architecture: Cloud Aware Application Development Ecosystem

```
node0# sudo apt-get install salt-api
node0# sudo apt-get install salt-master
node0# sudo apt-get install salt-minion
```

Now that you have salt installed on node0 (master node). Go to the master configuration file `/etc/salt/master` and add these lines.

```
file_roots:
  base:
    - /srv/salt/

pillar_roots:
  base:
    - /srv/pillar
```

There should be several things that are in the `/etc/salt/master` file commented out.

Get the fingerprint of the master node

```
node0# sudo salt-key -f master.pub
```

Save this string it will be used in the configuration of the minions.

Install Salt Minion on Node[0-4]

```
node1# sudo apt-get install salt-minion
```

Now edit the `/etc/salt/minion` file to contain the following

```
master: node0
master_finger: "Put output of 'salt-key -f master.pub' here"
```

Get things running

On node0 start the salt master as root in the foreground

```
node0# sudo salt-master
```

or in the background

```
node0# sudo salt-master -d
```

On node[0-4] start the salt-minions

```
node1# sudo salt-minion
```

Architecture: Cloud Aware Application Development Ecosystem

or in the background with the -d flag

```
node1# sudo salt-minion -d
```

now go back to node0 and accept the minions into the salt stack

```
node0# sudo salt-key -A
```

Now you can test and see if salt can see all of the nodes

```
node0# salt "*" test.ping
node0:
    True
node1:
    True
node2:
    True
node3:
    True
node4:
    True
```

1. Configure Salt states
2. Configure Salt Pillar
3. Download Salt Formula for CAADE

Install Gluster

Install Gluster on each of the nodes (node[0-4])

```
node0# sudo apt-get update
```

Install GlusterFS package using the following command.

```
node0# sudo apt-get install -y glusterfs-server
```

Start the glusterfs-server service on all gluster nodes.

```
node0# sudo service glusterfs-server start
```

Create Volumes for Gluster to use

Architecture: Cloud Aware Application Development Ecosystem

This assumes that you already have drives that have been mounted.

```
sudo mkdir -p /data/gluster
sudo mount /dev/sdb1 /data/gluster
```

Add an entry to /etc/fstab for keeping the mount persistent across reboot.

```
echo "/dev/sdb1 /data/gluster ext4 defaults 0 0" | sudo tee --append /etc/fstab
```

Now attach all of the nodes to each other. Go to node0 and type the following.

```
node0# sudo gluster peer probe node1
node0# sudo gluster peer probe node2
node0# sudo gluster peer probe node3
node0# sudo gluster peer probe node4
```

Now you can add volumes to the gluster cluster

```
node0# salt "*" cmd.run "mkdir -d /data/gluster/gvol0"
node0# sudo gluster volume create gvol0 replica 2 node1:/data/gluster/gvol0 node2:/data/gluster/gvol0
node0# sudo gluster volume start gvol0
node0# sudo gluster volume info gvol0
```

Mount Gluster Volumes on all of the nodes.

Now you have created a volume and now you can access it on all of the nodes by mounting it.

```
node0# mkdir /mnt/glusterfs
node0# mount -t glusterfs node1:/gvol0 /mnt/glusterfs
```

To make the mount permanent across reboots you need to add it to the fstab

```
node0# echo "node1:/gvol0 /mnt/glusterfs glusterfs defaults,_netdev 0 0" | echo tee --append /etc/fstab
```

Additional information can be found [here](#)

Docker Swarm

There is a great blog on how to generically set this up [here](#).

1. Using Salt Stack install
2. Test Docker Swarm Installation

Continuous Integration & Delivery

CICD

CICD is a subsystem of caade that is implemented by an existing CI/CD service that is available today. Examples of CICD systems that can be used are Jenkins, Bamboo, TravisCI, etc...

Use Cases

- [Test Application](#)
- [Test MicroService](#)

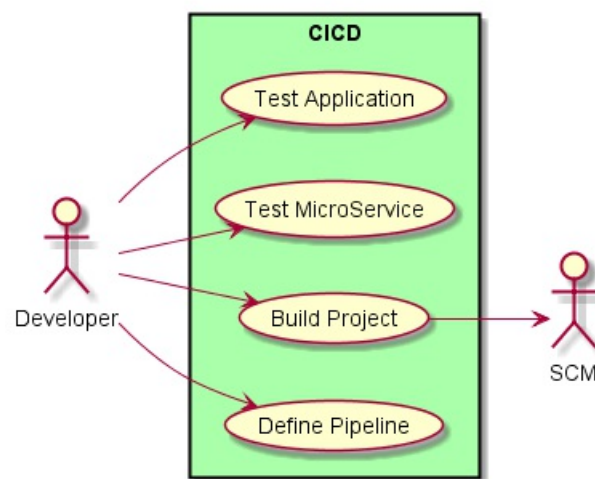


Figure: CICD UseCases

Actors

Users

- [Developer](#)

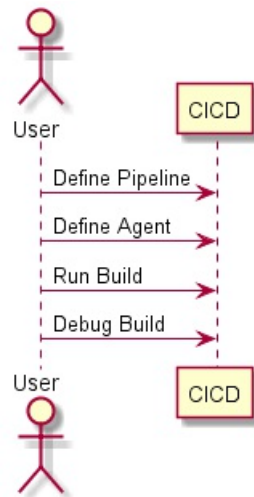


Figure: CICD UserInteraction

Uses

- [CICD](#)
- [Dev Cloud](#)
- [Local Cloud](#)
- [Production Cloud](#)
- [Test Cloud](#)

Interface

- CLI - Command Line Interface
- REST-API
- Portal - Web Portal

Logical Artifacts

- Agent - Agent running in the different clouds that perform builds for a Project
- Build - Build Stages of a pipeline for a project.
- Pipeline - Pipeline that defines how a project is built, test, and deployed
- Project - Project that contains the application and microservices

- Stage - Stage of builds defined in the pipeline.

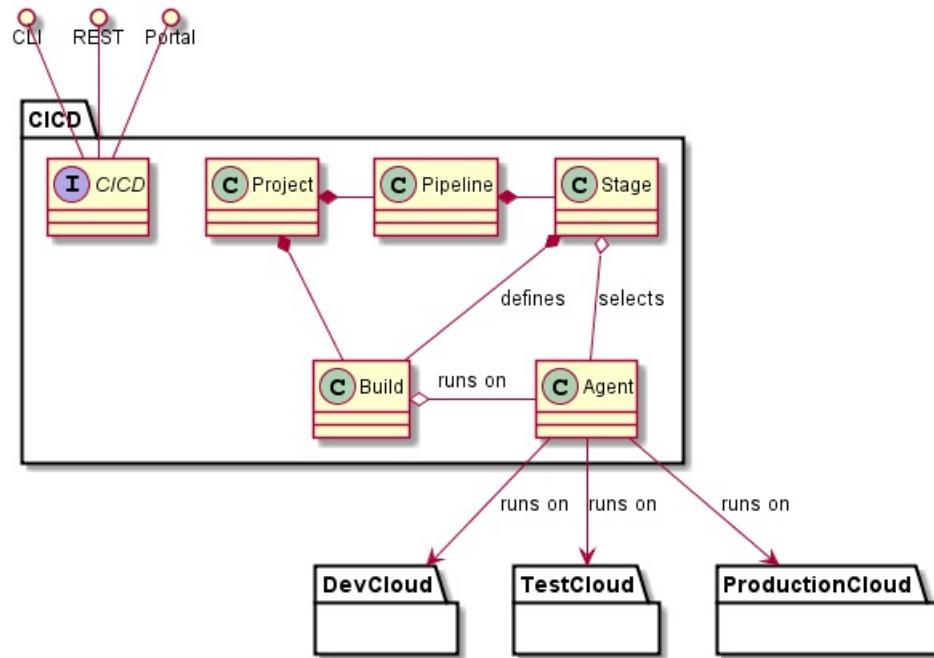


Figure: CICD Logical

Activities and Flows

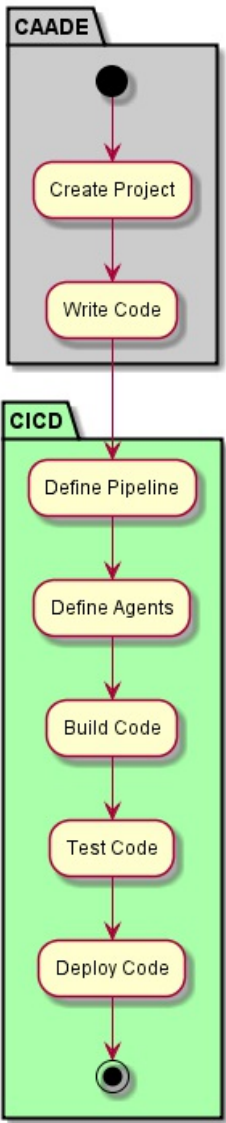


Figure: CICD Process

Deployment Architecture

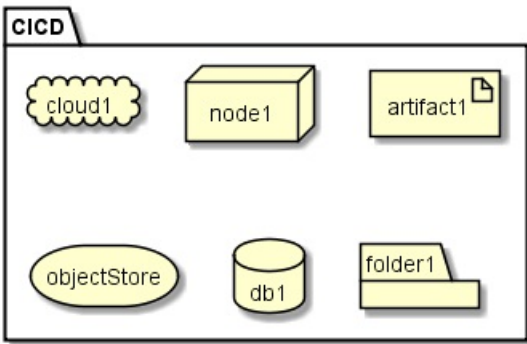


Figure: CICD Deployment

Physical Architecture

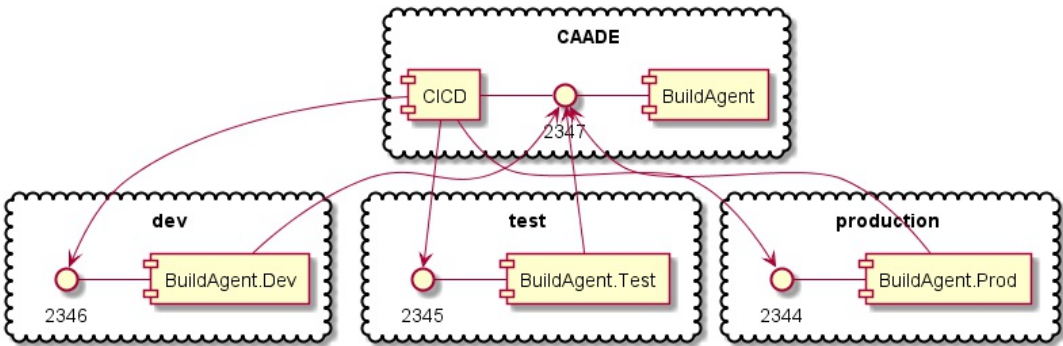


Figure: CICD Physical

Registry

Registry

Registry is a subsystem of caade ...

Use Cases

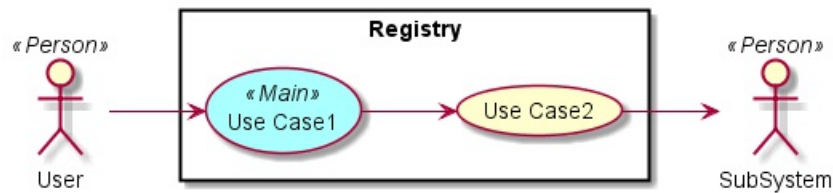


Figure: Registry UseCases

Actors

Users

- [User](#)

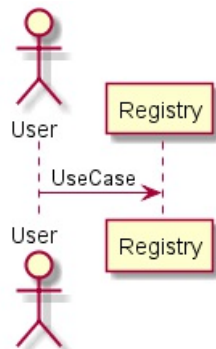


Figure: Registry UserInteraction

Uses

- [SubSystem](#)

•

Interface

- CLI - Command Line Interface
- REST-API -
- Portal - Web Portal

Logical Artifacts

*

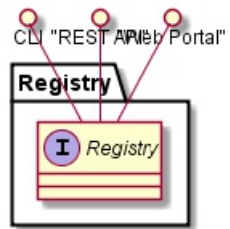


Figure: Registry Logical

Activities and Flows

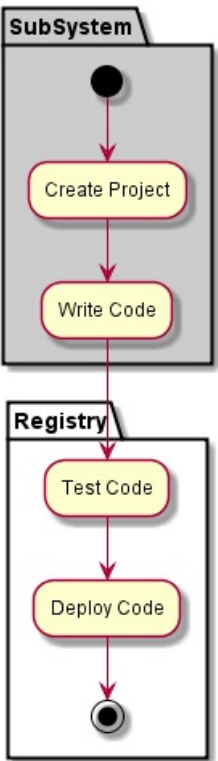


Figure: Registry Process

Deployment Architecture

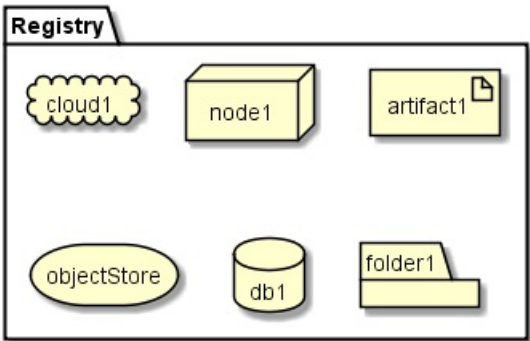


Figure: Registry Deployment

Physical Architecture

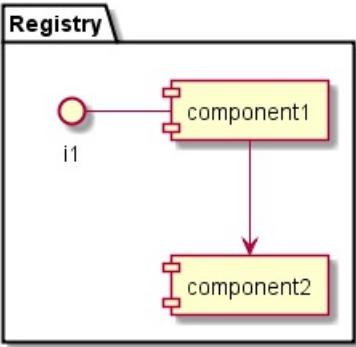


Figure: Registry Physical

Local Cloud

LocalCloud

LocalCloud is a subsystem of caade ...

Use Cases

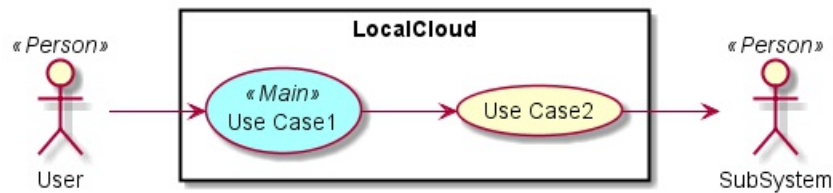


Figure: LocalCloud UseCases

Actors

Users

- [User](#)

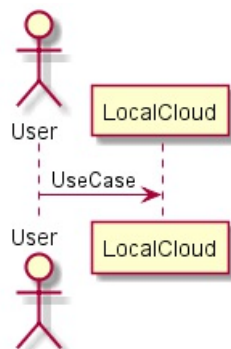


Figure: LocalCloud UserInteraction

Uses

- [SubSystem](#)
-

Interface

- CLI - Command Line Interface
- REST-API -
- Portal - Web Portal

Logical Artifacts

*

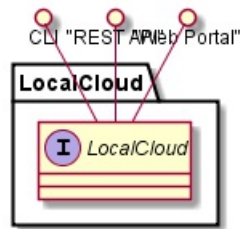


Figure: LocalCloud Logical

Activities and Flows

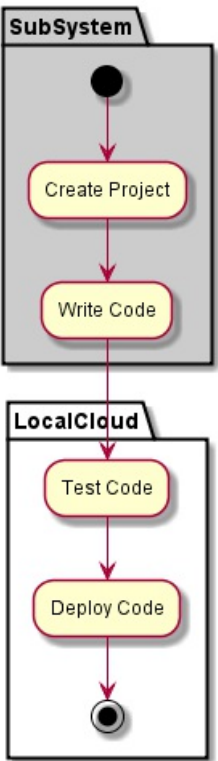


Figure: LocalCloud Process

Deployment Architecture

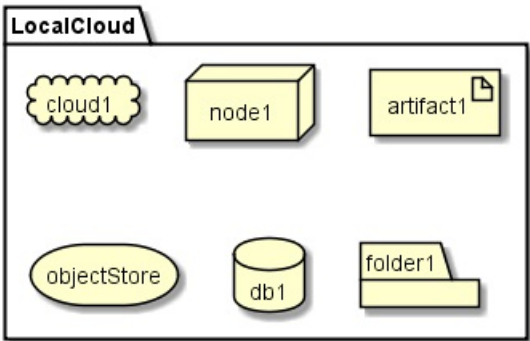


Figure: LocalCloud Deployment

Physical Architecture

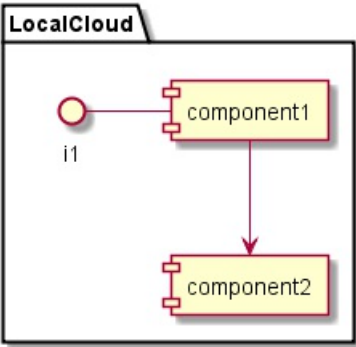


Figure: LocalCloud Physical

Dev Cloud

DevCloud

DevCloud is a subsystem of caade ...

Use Cases

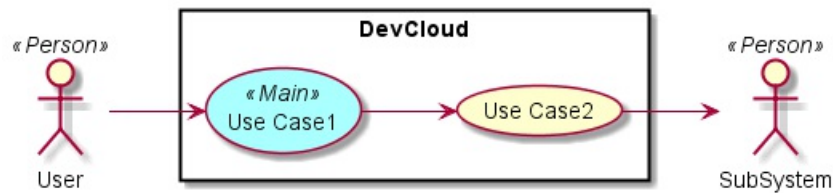


Figure: DevCloud UseCases

Actors

Users

- [User](#)

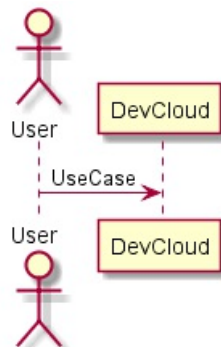


Figure: DevCloud UserInteraction

Uses

- [SubSystem](#)

•

Interface

- CLI - Command Line Interface
- REST-API -
- Portal - Web Portal

Logical Artifacts

*

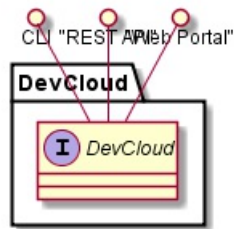


Figure: DevCloud Logical

Activities and Flows

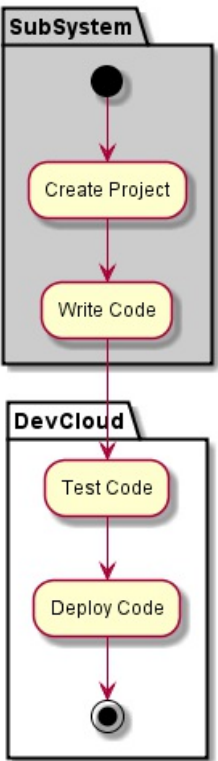


Figure: DevCloud Process

Deployment Architecture

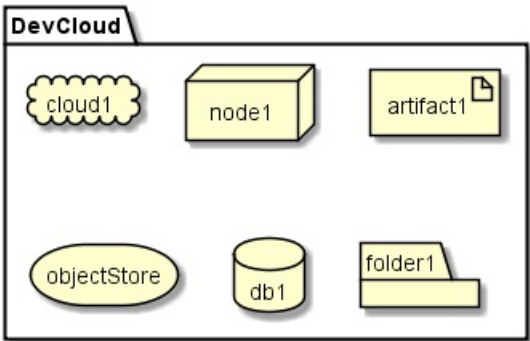


Figure: DevCloud Deployment

Physical Architecture

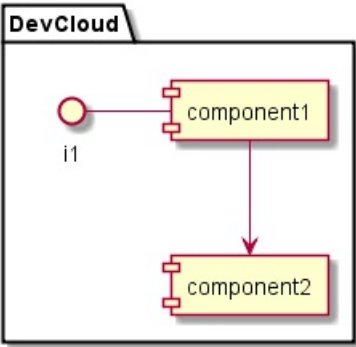


Figure: DevCloud Physical

Test Cloud

TestCloud

TestCloud is a subsystem of caade ...

Use Cases

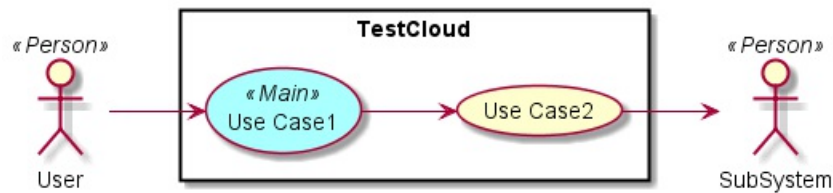


Figure: TestCloud UseCases

Actors

Users

- [User](#)

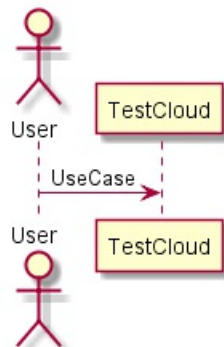


Figure: TestCloud UserInteraction

Uses

- [SubSystem](#)
-

Interface

- CLI - Command Line Interface
- REST-API -
- Portal - Web Portal

Logical Artifacts

*

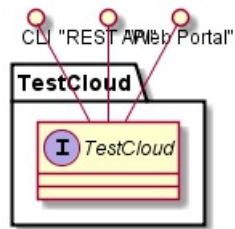


Figure: TestCloud Logical

Activities and Flows

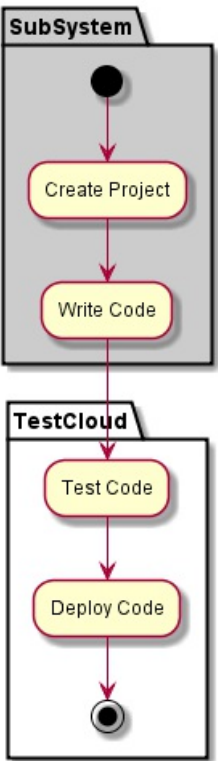


Figure: TestCloud Process

Deployment Architecture

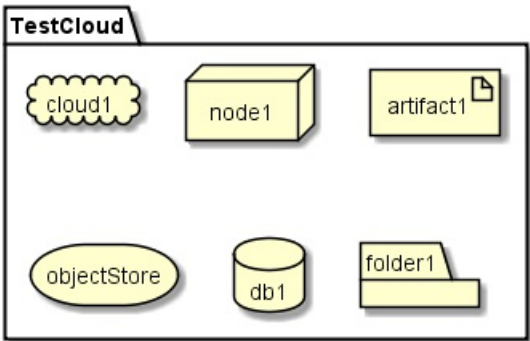


Figure: TestCloud Deployment

Physical Architecture

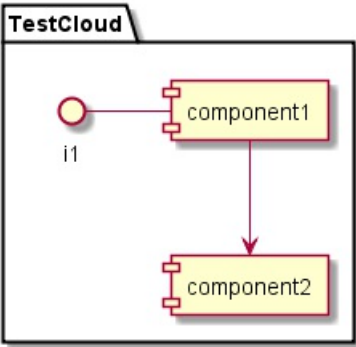


Figure: TestCloud Physical

Production Cloud

ProductionCloud

ProductionCloud is a subsystem of caade ...

Use Cases

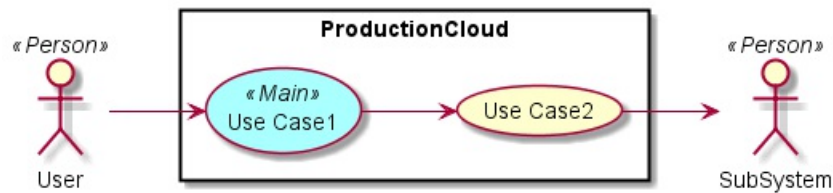


Figure: ProductionCloud UseCases

Actors

Users

- [User](#)

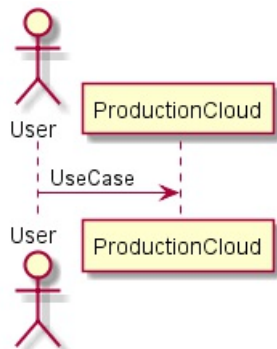


Figure: ProductionCloud UserInteraction

Uses

- [SubSystem](#)
-

Interface

- CLI - Command Line Interface
- REST-API -
- Portal - Web Portal

Logical Artifacts

*

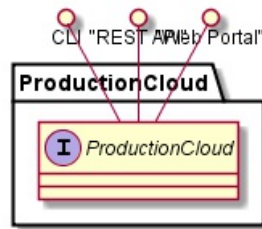


Figure: ProductionCloud Logical

Activities and Flows

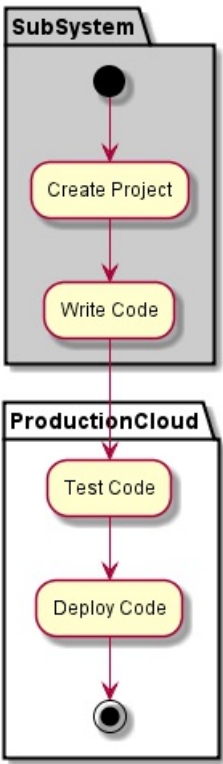


Figure: ProductionCloud Process

Deployment Architecture

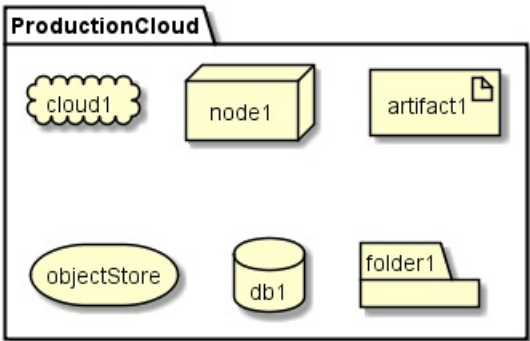


Figure: ProductionCloud Deployment

Physical Architecture

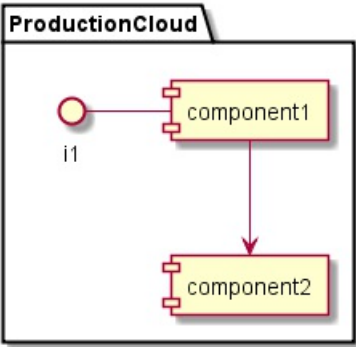


Figure: ProductionCloud Physical