Examples and tutorials

1. Examples
2. Configuration tutorials

1 Example

In the folder "examples" of SALSA source folder, there are a number of simple examples. This section introduces and explains about those examples. The examples are organized from the simple to complex ones, which can be used by modifying parameters and deployed by a SALSA service.

SALSA retrieves a TOSCA file as an input, which specifies the application structure. Please refer to the UserGuide document to have more details. With each example bellow, when submitting the TOSCA to SALSA, several virtual machines will be created and the application is registered and managed by SALSA.

1.1 Example 1: Deploy one virtual machine (VM)

This TOSCA contains only one Node Template with the id of SimpleOS. Inside the NodeTemplate, we describe a list of property as following:

- **Cloud provider**: The provider is divided into 2 part: site and platform. For example, dsg@openstack specifies the OpenStack cloud which is hosted at site name "dsg". The configurations of the cloud providers are specify in the user's configuration file. Please refer the UserGuide document for this.
- **Instance type**: A string to specify the instance type of the VM. The format of the string and instance name are depending on the specific cloud provider.
- **Base image**: The id of the image
- Packages: the default packages which will be install on top of the VM.

1.2 Example 2: Deploy an executable program on top of VM

This TOSCA contains 2 Node Templates:

- The HelloWorld node: The first node with id of "helloWorld" is a BASH script. The location of the BASH script is defined in the ArtifactTemplate.
- The VM node: similar to the VM in example 1.
- The HOSTON relationship: There is a RelationshipTemplate with type of "HOSTON", that defined the "helloWorld" node will be run on the VM node.

1.3 Example 3: Deploy with docker

This example is similar to the example 2, but we add one more NodeTemplate with type "docker" in the middle of HelloWorld and SimpleOS node. We have two HOSTON relationships which show that the HelloWorld artifact is deployed on docker and the docker is deployed on the VM SimpleOS.

1.4 Example 4: Deploy with Tomcat

This TOSCA contains 3 NodeTemplates:

- The web application named PoliceApp. The artifact is a .war file, which is defined in the ArtifactTemplate.
- The Tomcat NodeTemplate with type is "tomcat". SALSA will automatically recognized and get the deployment information from its metadata to deploy a Tomcat server.
- The VM NodeTemplate is similar to previous examples.

1.5 Example 5: Complex topology of M2MDaaS

This TOSCA contains the description of the M2MDaaS with two topologies, one for the DataEnd and the other for the EventProcessing.

1.6 Example 6: Multiple cloud configuration

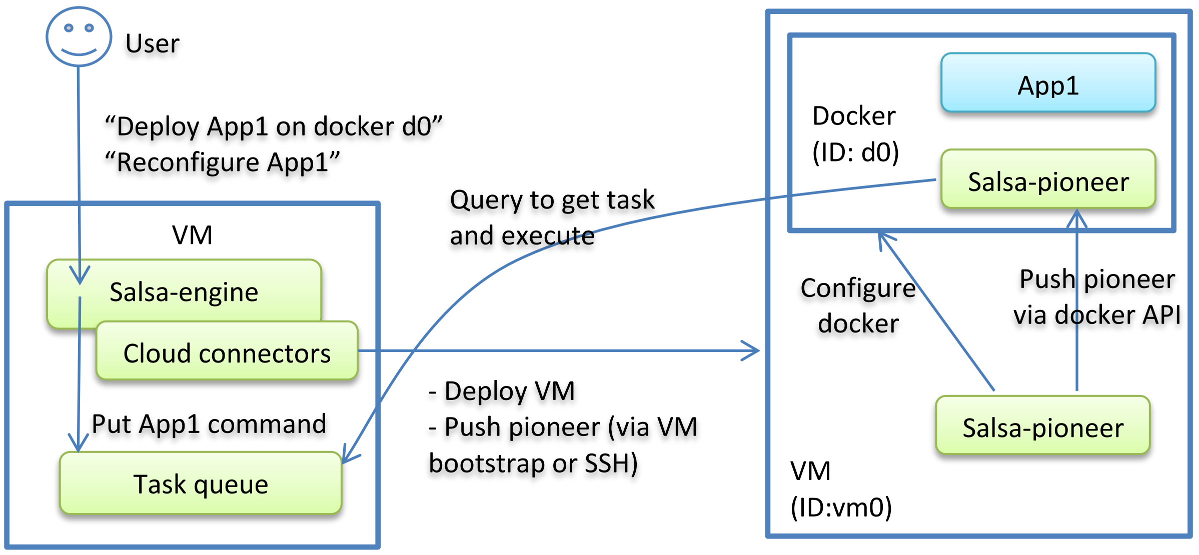
One application can be deployed across multiples cloud. In this example, we deploy a MQTT server (the ActiveMQ) on Flexiant infrastructure and a number of sensor clients to push data to the MQTT on OpenStack. As SALSA already supports deploying multiple clouds, users need to specify the properties for the VM of each of the components.

2 Configuration tutorials

This section is going to answer several questions which then guide users in using SALSA to satisfy the requirements of the complex configuration.

Question: How do users deploy something on a VM which are already running or when a container is already running, etc. How does SALSA work with multiple stacks?

Answer: Current architecture of SALSA enables to deploy and dynamic configure at the infrastructure and application level. On the infrastructure level, a lightweight client called salsa-pioneer is injected.



Whenever a salsa-pioneer is inside the VM or OS container (like docker) which refer to the infrastructure level, it will continuously queries for tasks that assigned to it (see figure above). There are following cases that the VM is already running:

- VM, container and application are defined in the same TOSCA at the beginning: they can be deployed/configured at any time as pioneers are always ready.

- A none-managed VM is bound to the service at runtime, which no pioneer is inside, we need to run an pioneer as well as the configuration file inside. This can be done by salsa-engine via RESTful which will try to use SSH to connect to the VM, or manually install/run the pioneer.

Question: Users may also have an agent to be pre-deployed in a gateway/VM which will download other code and do the deployment, how you SALSA plan to reuse it?

Answer: In order to use application-specific configuration tools, SALSA will need an adapter which implement the *salsa-pioneer instrument interface*. Please check:

https://github.com/tuwiendsg/SALSA/blob/master/salsa-core-pom/salsa-pioneer-vm/src/main/java/at/ac/tuwien/dsg/cloud/salsa/pioneer/instruments/InstrumentInterface.java

When recognize the type of the artifact, SALSA will forward the request and necessary information into the instrument module, which eventually invoke the external user-defined configuration module.

Question: How a new deployment algorithm can be implemented and configured - so i can have different algorithms (and i see which one is the best)

Answer: During the configuration, SALSA contains two point to apply algorithms

- Before the configuration: SALSA refines and enriches the application description (which is TOSCA). In current implementation, all of the algorithms are developed in following URL and will be called at the new orchestration.
  https://github.com/tuwiendsg/SALSA/tree/master/salsa-core-pom/salsa-engine/src/main/java/at/ac/tuwien/dsg/cloud/salsa/engine/smartdeployment
  TODO: Refactor the above package to support the attachment and selection of algorithms.
- During configuration: SALSA contains the algorithm for orchestration the configuration and for placing components at runtime which are implemented in the class:
  https://github.com/tuwiendsg/SALSA/blob/master/salsa-core-pom/salsa-engine/src/main/java/at/ac/tuwien/dsg/cloud/salsa/engine/impl/SalsaToscaDeployer.java
  Currently, the orchestration is design by parallel threads which can change to work-flow style or queue-based style. The placement is based on checking maximum instance number.
  TODO: Refactor to extract the algorithms outside of the routine.

Question: How to combine different deployment and configuration algorithms to have a new one.

Answer: Currently SALSA do not support multiple algorithms yet. We vision that when SALSA is able to deal with multiple algorithms, which apply for different part of the service (e.g. on service unit, on one topology or whole application), user can annotate the application specification to guide SALSA.

Question: how to have configuration processes - which are application/system-specific - who writes what and salsa will do what

Answer:

With the cloud providers, SALSA supports basic operations which implement the cloud-connector interface. The interface can be extended to support more capabilities from providers in general.

With the application level, SALSA support to defined custom actions can be invoked at runtime. In current implementation, we support user to execute custom script at runtime. By this, users provide a set of scripts to execute various configuration capabilities and specify with the service unit. At runtime, these actions can be executed via RESTful API with action name as input. For example, here is a piece of specification.

```
<tosca:Properties>
      <MappingProperties>
            <MappingProperty type="action">
                 <property name="start">sudo service myService start</property>
                 <property name="stop">sudo service myService stop</property>
                 <property name="reconfig">./reconfig.sh</property>
                 <property name="undeploy">./uninstall.sh</property>
            </MappingProperty>
      </MappingProperties>
</tosca:Properties>
```

Please refer to the UserGuide for more detail on “action” property.