SALSA Features

Main Features	Description	Limitation and TODO
Automate the cloud resource provisioning	SALSA can connect to different cloud system to manage VM provisioning.	The capabilities are fixed with the cloud connector interface with limited operation (create, remove, getInfo...). **TODO**: The interface can be extends to capture more functionalities from cloud providers.
Multiple stacks deployment	The configuration of infrastructure, containers and applications stacks are separated, support fine-grained configuration.	-
Runtime configuration on multiple stacks	The configuration capabilities of stacks and service units are exposed to SALSA API to invoke at runtime.	-
Wire configurations of service units	Support two service units to share parameters during their configurations.	At the implementation level, need to test and enable custom parameters. Currently just test to transfer IP. **TODO**: revise the API that support application to set/get shared parameters.
Centralized orchestrating the configurations	Single salsa-engine stay for coordinating the configurations, sharing parameters and exposing capabilities.	Reduce the performance for configuring highly distributed cloud services because of data transmission and service call. **TODO**: several components of central salsa-engine can be moved to the local cloud.
Manage configuration dependencies	One configuration can trigger other configurations.	Just support default actions for deployment (deploy, undeploy, start, stop). They can be executed at runtime, but not support custom actions yet. **TODO**: introduce the RuntimeConfiguration relationship, which support custom configuration dependencies.
Configuration states report	The configuration progress is reported via states and the result as done or error.	SALSA does not manage the service unit runtime state, e.g. if the service is stopped by users, SALSA does not recognize. **TODO**: introduce in the interface to get the runtime information of different service type, then some adapters to check. E.g. VM via cloud API, system service via “service [name] status”, webservice via connection availability, or from other tool like MELA. **TODO**: can the service have custom states?

Implementation Features	Description	Limitation and TODO
TOSCA parsing	Use TOSCA for describing	
Network topology independency	There is no need the connection opened for the salsa-pioneer because it connects to salsa-engine to share info, get command queue, etc. E.g. components in private network or inside docker container can be configured.	Reduce the performance while salsa-pioneer checks the salsa-engine by a frequency. Also it requires salsa-engine to be public with salsa-pioneer. **TODO**: SALSA components can communicate via a message queue. Then salsa-engine can stay in the developer laptop.
Support docker configuration	Developer can provide custom Dockerfile or request for default docker container. Software stacks then can deploy on top of this.	Fully support docker with Ubuntu image. The limitation is the container must include Java (which is not with e.g. Busybox) to run the agent. **TODO**: cooperate with rGovOps provision agent to configure the IoT-like components.
Support default war artifact and Tomcat	Developer can define a war file and SALSA automatic configure Tomcat (by having in SALSA knowledge)	-
GUI and RESTful services	Show the configuration states and service topology, expose API as cloud service structure.	GUI not show the concepts clearly.
Integrate with rSYBL	SALSA expose APIs that is specific for SYBL and scale-in, scale-out capability	