ONAP Modern Approach to implement VF creating in SDC

SDC is ONAP visual modelling and design tool. It creates internal metadata that describes the resources used by all ONAP components at design time and runtime. SDC can select directory items from the management directory and design them through logical combination.

1. General background

   SDC manages four levels of resources:

   - Resources: A basic function that can be implemented entirely in software or software that interacts with hardware devices. Each resource is a combination of one or more virtual function components (VFC) and all the information needed to instantiate, update, delete, and manage the resource. The resource also includes information related to the license. There are three types of resources:

     - Infrastructure (cloud resources, such as computing, storage)

     - Network (network connection functions and elements); example: virtual network function (VNF)

     - Application (function of software application); example: load balancing function

   - Business: A structurally complete object containing one or more resources. Business designers create services in SDC, including instantiating, updating, deleting, and managing all the information required by the service.

The main output of SDC is a set of models, which contains descriptions of asset functions and instructions for managing assets. These models are stored in the SDC main reference catalog.

- SDC supports mirrored artifacts storage
- TBD

2. Cons

- Currently SDC is only concerned with composition design
  - Once SDC provide composition info, TBD
  - Since SDC acts as a tool in design mode, in addition to the need to meet the requirements of design mode in function, it also needs to take into account the availability and ease of use of the operating state or even southbound. There is room for optimisation in this part of the SDC.
- Current implementation is inconvenient
  - The current SDC background storage uses the Cassandra database.
  - When uploading more than 10M file size, it will fail to upload.

3. Ideas

- Extend the current SDC so that it can save the image to the VF, thus facilitating the subsequent instantiation of the VF.
- Due to the actual application scenario, when creating the VF, the image package required by the VF needs to be compressed in the Csar package, which is convenient for subsequent Instantiate.
- The SDC architecture uses the Jetty server as an application server and is developed in a way that separates front and back ends.
- The Catalog-UI provides all the static web content required by the SDC front-end interface display (homepage, model upload, model warehouse) and all resources required by the GUI.
- Catalog-FE acts as a proxy for REST API requests from the GUI and forwards requests from the foreground to the Servlet of the BE

in the background. Each request from the GUI will be passed to the Jetty front-end server before being executed.

- Catalog-BE and Onboarding-BE contain all the business processing logic of SDC backend, including model upload, service creation and import, VF creation and import and other functions.

- SDC uses two storage components: Elastic Search and Cassandra:

  - Elastic Search is used to index audit data received from different operations of SDC, and then Kibana can be used to analyse this information.

  - Cassandra is used to store audit data, Artifact data and data models (VF, Service, VFC, etc.).

4. Implementations

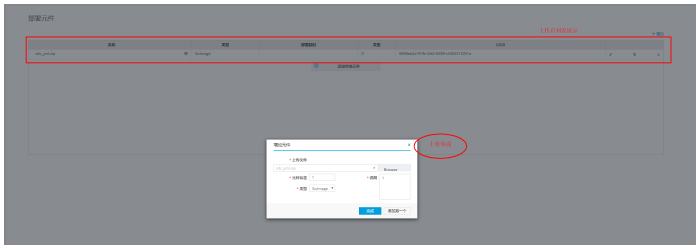
- Added new upload types SwImage (image file type) and SwImageDesc (image description file used to describe all information related to all images contained in the VNF package) in the deployment component interface of VF.

- The uploaded image is currently stored locally.

- The Docker volume mapping method is stored in the server deployed by the SDC, and then the current VF and mirror mapping relationship is maintained in the current sdcartifacts relationship table.

- When downloading the VFCsar package, the uploaded image information is obtained according to the VF Component ID and then packaged into the downloaded Csar package.

5. Conclusion



6. Future Roadmap

TBD