Service-types overview

Introduction

Service-types are used to create automated changes. The designer uses these Service-types to specify in fairly high detail how specific changes are to be executed using the method and design as he requires. A Service-type manipulates step-by-step the various objects in the NetYCE 'abstract network' to create the desired change.

Such Service-types only modify the intern NetYCE database objects and have no other effect than the manipulation of data and their relations. Once a Service-type is completed, the operator than can select or create the appropriate change jobs to activate the modified configuration to the network. They serve to standardize changes and tasks in preparation of the actual configuration change in the same way as the templates standardize the configuration lines that make up a configuration.

Service-types are created and tested once - usually by a designer and are then executed many, many times by the engineers or operators. They provide a comprehensive method to automate simple or complex tasks in a standardized way without coding, fully vendor independent, but design specific. Service-types allow you to enforce to the design rules of a specific network environment.

When creating a Service-type, the designer must first specify to which Client-type it applies - Client-types roughly match a distinct Network Architecture. Then he also needs to specify to which Service-class the Service-type will belong. A Service-class groups various Service-types that belong together in a class. The Service-classes are made available by assigning them to a Site-type (location) in a Client-type.

This way, the designer can create all kind of "services" that are only available on specific location types but can be implemented in a few but well-defined ways.

Furthermore, each Service-type has Service-tasks associated. Normally a Service-type has one Service-task labeled "Create" to do just that - create the service for the first time. Then other Service-tasks can be created within that Service-type to perform specific modifications for extending the service, migrating the service, cleaning it up or any other task that benefits from automation.

The Service-types are part of the modelling "hierarchy" that NetYCE uses:

```
Modeling:
---------
Client-type
      Service-class
          Service-type
               Service-task
```

The levels of the above hierarchy match the corresponding levels found of the abstract network. The Modelling represents the architecture, design and tasks of a network. They are defined once. the abstract network represents the "life' network using this architecture, design and tasks. Many clients, locations, service, nodes, etc will be created and maintained using the definitions found in the modelling tree.

```
Abstract network:
-----------------
Client              (client_type)
    Site            (site_type)
        Service     (service_class and service_type)
             Node   (node_class and node_type)
```

Rollback on error

In the chain of commands, something can go wrong. In that case, all modifications on the database are rolled back and undone, so that you don't need to clean them up. This a new feature, available from version 8.2.1

Example

To clarify the use of Service-types, a simple example could be helpful.

Assume a Managed Lan network (the client-type) where the network consists of large or small locations. The small locations have a non-redundant aggregation core, the larger a redundant set of nodes as core. The “services” provided at these two site-types are ‘core’ and ‘access’ device layers. These two form the service-class names available on these sites.

The design stipulates that the cores are implemented using a single switch type but in two variations: single and redundant. So this leads to two Service-types under the ‘core’ service-class. Likewise, the ‘access’ switches have the same two variations, but are also available in 24 and 48 ports. That leads to four Service-types for the ‘access’ service-class.

When an operator creates a new client in the Managed Lan client-type, that client refers to a new customer using that Managed Lan design. there can be hundreds of them or just one or two, it all depends on the type of network and the scope of the network(s). Each Managed Lan client will have a bunch of locations, all being either small or large.

The operator simply selects the client and the desired location and chooses to “add a service”. He is then prompted with a choice for the service-class to add (usually in the form of an uploaded network diagram). And, after the selection a subsequent prompt to select the Service-type to execute.

In case one of the ‘access’ Service-types is to be executed, it is then responsible of creating service containers and its node(s) and subnets as per the site-type design. But the Service-type can also continue to add all the remaining features of the design: linking the access devices up to the core device(s), allocating IP subnets from the IP-plan, assigning them to the access ports and uplinks/downlinks, creating the L3 vlans on the core, setting port ACL’s and a host of other details if the design demands.

Bottom line is, that Service-types can be used to define and assign all the variables and their contexts that are needed to make up an entire multi-node service configuration. It is the task of the templates to turn these variables into vendor specific CLI configuration lines (or other formats).

> A complete reference to all service type commands is listed in Service-type syntax