Generation of REST API and DSC Data Models

The DSC REST API is generated automatically based on the data model using DSC REST API Generator (DRAG) utility. The DSC data model is an XML file with nodes describing Predix Experience (Px) containers and types. The data model is parsed to create the REST API definitions or entry points.

The schema of the XML files is based on the application's DEF files and uses the YANG data modeling language (as defined in RFC-6020) for descriptions and can exist in different formats such and YIN and YANG. These formats are interchangeable and have the same information. The YIN notation enables YANG data model to be represented in XML and use a rich set of XML-based tools for data filtering and validation, generation of code and documentation.

During the nightly build, the DSC data models are generated as YIN files for every supported application. After the data models are generated, the DSC REST API Generator (DRAG) processes the data models and produces a web application (DRAG instance), which serves the REST API.

Upon receiving an HTTP request, the DRAG instance converts the incoming HTTP method (GET, POST, PUT, or DELETE) and URL into a TL1 command.

There is one dedicated TL1 socket connection for each user/password combination.

- GET (Read): The REST API interface can read configuration data. When a request is sent to read a list (several objects of the same px type), the REST API returns ALL list elements.
- POST (Create): The REST API interface can create new objects provided that the object is "creatable". When creating a new object, a parent must already exist. Only one object of a particular type can be created per REST API call. children will not be created. For example, the first REST API call can create a MTP3 NA, then a second REST API call is sent to create a VNode in this NA. This is different from the "bulk" XML provisioning, where a MTP3 NA can be created with all the children.
- PUT (Edit): The REST API interface can update objects. The edit operation can modify attributes directly. If the modification requires a deactivation or some other action, the response will return an error with a text description that can be used to understand the problem. If the object requires deactivation, perform the required action by issuing another HTTP PUT API call, then repeat the original request.
- DELETE (Delete): The REST API interface can delete objects. When deleting an object, no implicit deactivation is performed. If deletion requires a deactivation, the response returns an error (HTTP response error code accompanied with description). In another situation, an object might not be deleted because it has child objects. Delete the child objects with other REST API calls before deleting the parent object.

The objects which can be operated upon (POST/PUT/DELETE) are subsets of objects that can be read (GET). For example, in the case of the DSC application, an API can read DSC instances in response to a read request but can not create a DSC instance in response to create request. In other words, for each object, the REST API does not necessary support all four HTTP methods.

For more information about the HTTP methods, see HTTP Methods.

The DSC data models contain all the definitions required for the provisioning. The following table summarizes supported applications and corresponding data models.

> The following table does not include the Alarm Synchronization or the Inventory Management because their resources are not automatically generated into YIN files and do not appear in the REST API Reference Guide. Instead these features use special customized resources. For more information on these features, refer to Alarm Synchronization and Inventory Management.

Table 1: Data Model Files

Application	Model
Number Portability Gateway (NPGW)	sonusnpgw.yin
Diameter Signaling Controller (DSC)	sonusDSC.yin
InterWorking Function (IWF)	sonusiwf.yin
Intelligent Network Application Protocol Gateway (INAPGW)	sonusinapgw.yin
Message Transfer Part Level 3 (MTP3)	sonusMTP.yin
Signaling Connection Control Part (SCCP).	sonusSCCP.yin
Gateway Screening and MSU Tracing (GWST)	sonusGWST.yin
Global Title Translation (GTT)	sonusGTT.yin
Signaling Gateway (SG)	sonuysSG.yin
Point Code Emulation (PCE)	sonusPCE.yin
L4 Converter (L4CVTR)	sonusL4Cvtr.yin
Integrated Monitoring Feed (IMF)	sonusimf.yin
Message Transfer Part Level 2 (MTP2)	sonusIOHW.yin
Load Balancer (LB)	sonusLB.yin