October 03, 2019

METADATA DRIVEN USER INTERFACE GENERATOR BASED ON DEVICE TYPE

Jasvinder Bhasin

Ganesha Shankara Bhat

METADATA DRIVEN USER INTERFACE GENERATOR BASED ON DEVICE TYPE

AUTHORS:
Jasvinder Bhasin
Ganesha Shankara Bhat

ABSTRACT

Techniques are described herein that provide an innovation in which each device type within a system may receive its own metadata in order to draw (e.g., display) and populate user interface (UI) elements. Different device types and their corresponding UI elements (e.g., tabs, tables, columns, etc.) are visible from one devices' list page. This enables the system to scale in multiple device categories (e.g., routers, gateways, endpoints, etc.). The system can be extended to include other device categories that a current product may not be managing.

DETAILED DESCRIPTION

Different devices within a system may have different information associated therewith that may be shown or displayed in various formats such as text, lists, tables, etc. One mechanism that may be utilized to achieve showing or displaying information for a device may involve the user interface for a device generally predefining properties such as columns, names, labels, etc. Another mechanism that can be utilized may involve the user interface obtaining the predefined properties from a backend. However, such mechanisms may be hard to maintain. Further, it may be difficult to handle different versions using such mechanisms.

Current network management systems (NMS) do not have a metadata driven approach for network devices; therefore, creating user interfaces (UIs) for lists of different devices can be a laborious task. Current network management systems also generally do not allow for all the different types of devices being managed to be shown in a single page.

Techniques proposed herein provide innovation involving a system in which metadata that describes the properties of user elements can be parsed and stored in a database by a metadata processor. A request engine can fulfill a UI request containing

device type by retrieving the metadata and device contents from the database. The device contents may include attributes of the device. A rendering engine in the UI that receives the metadata for a particular device along with the device's data from the request engine can display corresponding UI elements with data populated therein.

Each device type may include multiple attributes in the form of a JavaScript Object Notation (json) stored in the database. At the time of start-up of an application, metadata can be parsed and stored in the database. If metadata for a device type already exists in the database and there are any modifications, then changes can be updated in the database.

Metadata properties can include any information such as, for example, whether a column is sortable and/or capable of being hidden, a name of a column header, a default width of a column, combinations thereof, and/or the like. If column data contains link data then a Hypertext Reference (href) can also be included in the metadata properties.

Each time a user selects a particular device type that is supported by the system, a request is made to the request engine. The request engine will be responsible for gathering the metadata and device contents and bundling them together in one single response content.

The rendering engine is responsible for rendering UI elements based on metadata received from the request engine. For example, the rendering engine can make a decision whether data may be displayed in a tabular format or, if multiple tabs' data sets are found in the metadata, then a table within the tab format may be used. Example screenshots for different UI elements that may be rendered using techniques proposed herein are shown below in Figures 1 and 2.

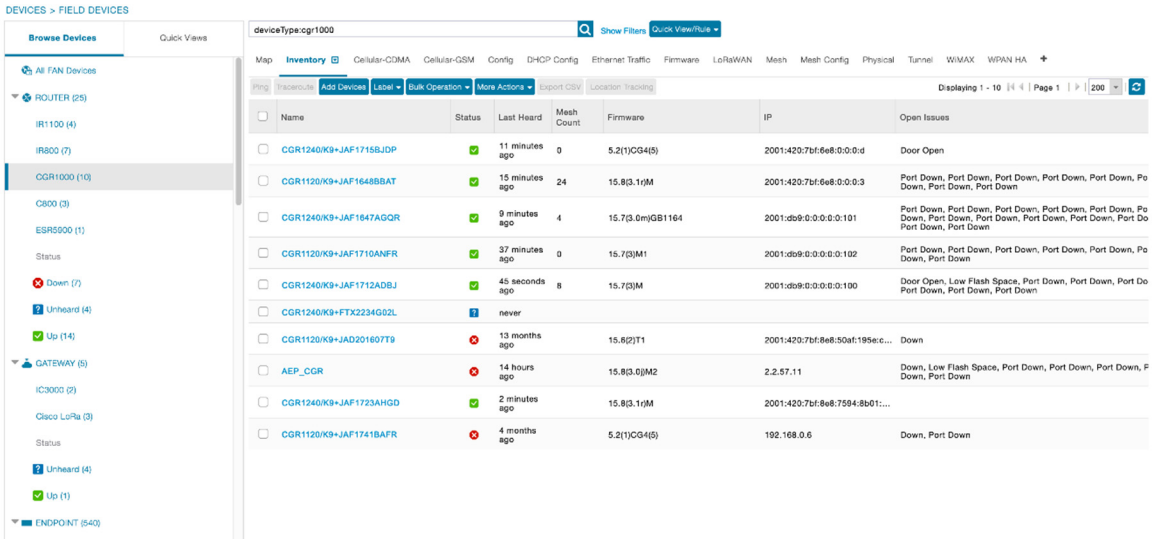

Figure 1

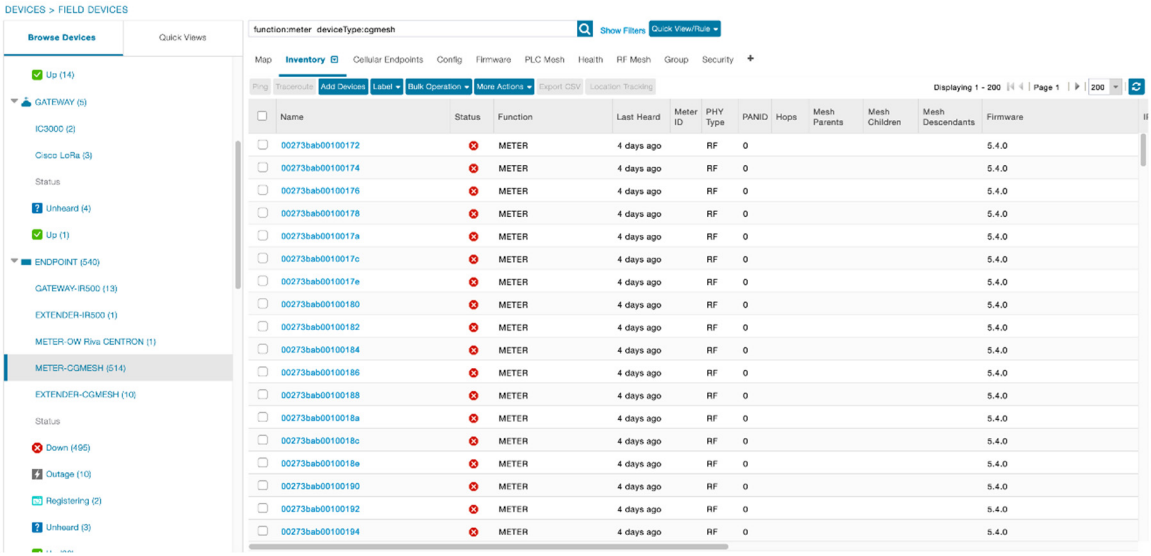

Figure 2

Beyond rendering UI elements based on the metadata, a search component may also utilize the same metadata to generate searchable fields along with their UI labels and categories for a selected device type. UI rendering for the search component may utilize the metadata defined types such as, for example, date, string, number, etc. In some implementations, searches can be saved and restored. Example screenshots for different searchable fields that may be generated and rendered by the search component are shown below in Figures 3 and 4.

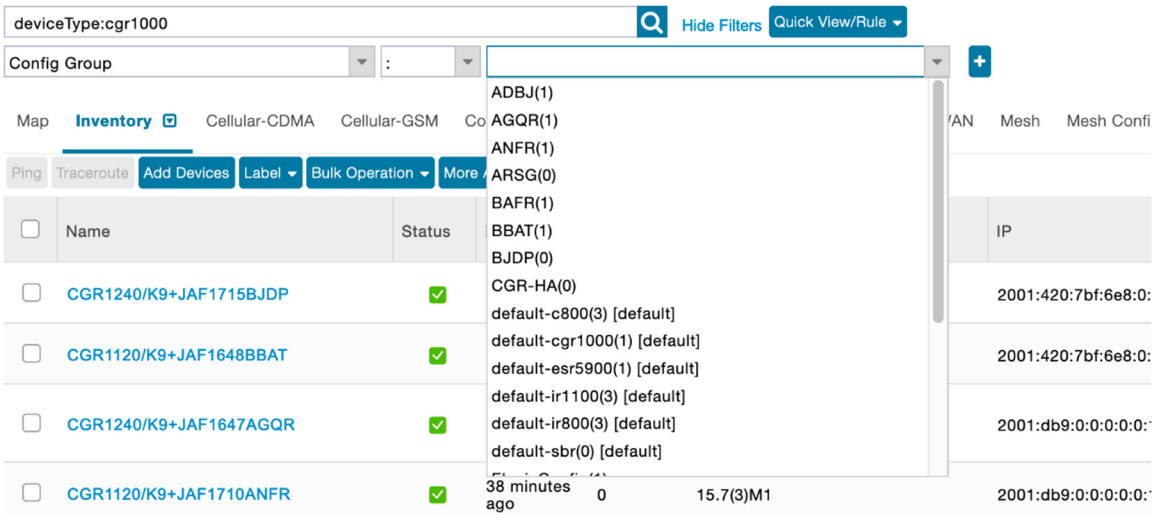

Figure 3

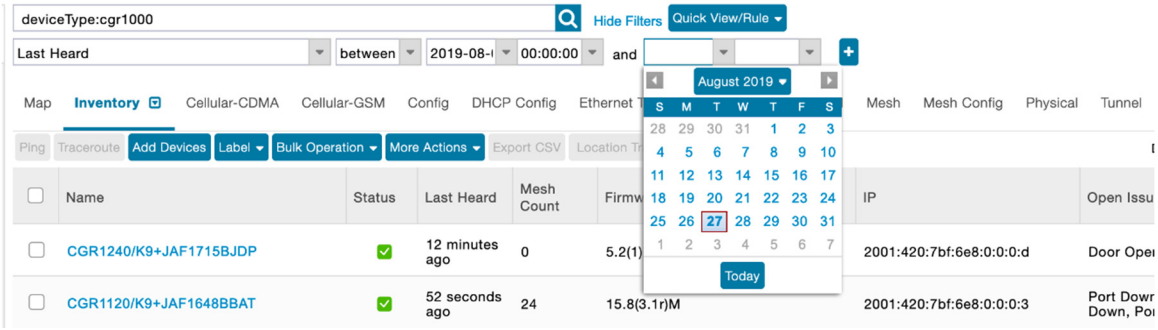

Figure 4

In summary, techniques presented herein provide an innovation in which each device type within a system may receive its own metadata in order to draw and populate user interface (UI) elements. Different device types and their corresponding UI elements (e.g., tabs, tables, columns, etc.) are visible from one devices' list page. This enables the system to scale in multiple device categories (e.g., routers, gateways, endpoints, etc.). In some implementations, the system can be extended to include other device categories that a current product may not be managing.