IAB NEMOPS Position Paper - Telefonica

The advent of model-driven protocols like NETCONF or RESTCONF, along with the YANG data modeling language, marked a breakthrough in the network management and operations area. These technologies have brought several advances both in configuring and monitoring the network, the two requirements for enabling a closed loop that can deliver the promised autonomous networks.

Traditional management methods based on vendor-specific CLIs have been superseded by Application Programming Interfaces (API) grounded on standard protocols and data models formally defined using the YANG language. With this, the first two levels of data interoperability have been achieved: i) the protocol interoperability is realized by means of standard protocols like NETCONF; and ii) the syntactic level of interoperability has been achieved with the wide adoption of YANG.

However, the success of the YANG language in the industry has created a tsunami of data models developed by different actors ranging from SDOs and network vendors to consortia like OpenConfig. The original plan of standard YANG data models “to rule-them-all” was deviated with vendor-specific extensions, competing standard models (e.g., IETF vs OpenConfig), or overlaps between data models.

Such heterogeneity of YANG data models has hindered the consumption and integration of YANG data, showing different data models that refer to the same concepts. This **Limitation #1** is precisely the goal of the semantic interoperability level, whereby applications must have a common understanding of the concepts. In this regard, the YANG language should evolve towards a grounding on formal knowledge representation to achieve the semantic interoperability level. Standards like Resource Definition Framework (RDF) and Web Ontology Language (OWL) from the Semantic Web may serve as reference in this area.

Following on interoperability, **Limitation #2** relates to the reuse of fragments among YANG data models. The “grouping” mechanism was introduced in RFC 7950 to this end, however, over time it has proved to work inefficiently across data models. Several additional mechanisms have been proposed like Schema Mount [RFC8528], Peer Mount [I-D.clemm-netmod-peermount] or YANG Full Embed [draft-jouqui-netmod-yang-full-include-02], however, they all focus on reusability at the syntactic level, i.e., YANG language. These mechanisms save time for data model developers as less code is written and duplicated, but still, the miss the key point of semantic interoperability. The evolution of YANG language towards a semantic language should unlock the reusability at the semantic level, where not just syntactic fragments are reused, but the concepts they refer to.

The YANG language was born with a hierarchical mindset, where there must always be a root and everything else hangs from it. Structuring network data in a hierarchical, tree-like shape

brings several benefits as seen in the success of JSON or XML. Data are easier to read and can easily iterated and processed using query languages like XPath. However, the complexity of networks keeps increasing, so they data that they produce (**Limitation #3**). This fixed hierarchical structure limits the modeling of complex scenarios where relationships within the data are important. This is the example of data models like SAIN [RFC9418], YANG Library [RFC8525], or network topology [RFC8345]. In this sense, graph structures provide a more flexible structure that can accommodate any kind of data.

Similarly, the complexity of the network not only has increased in the devices but in the network as a whole. In this sense, the realization of the autonomous networks requires AI/ML applications to access scattered data sources over the networks and combine their data. Such integration of distributed data silos over the network, not only calls for semantic interoperability, but also for mechanisms that can uniquely identify "things" in the network. YANG has proved to be a local-oriented language, where identifiers of things like "interfaces" where locally scoped (**Limitation #4**). This impedes the connection of data originated at different levels of the network, e.g., interface at the device level and interface at the network or service level. In this regard, approaches based on IRIs/URIs to uniquely identify these resources can enable a distributed integration and management of resources throughout the network.

Last, but not least, is the absence of a formal open-source community around the YANG ecosystem (**Limitation #5**). This has greatly limited the innovation of the ecosystem and the adoption of the language beyond the network industry. A good example is the lack of mature YANG libraries, with several efforts in different programming languages that have been abandoned or that do not keep the pace of the standards.

Another example is YANG Push, which was standardized several years ago and is envisioned by the industry as a key feature for network monitoring. Unfortunately, there are still no open-source implementations of neither servers nor clients that support YANG Push. Late efforts in the IETF Hackathon have aimed at improving the situation, but the ecosystem is far from having a strong open-source community (close to OpenConfig).

Moreover, maintenance and up-to-date documentation of the YANG models is a not agile process (**Limitation #6**), thus slowing down the adoption and general usage in the industry, impacting also interoperability and integration.

Some other aspects are also worthy to be highlighted. This is the case of the atomized or un-coordinated work on modeling different pieces or parts of the overall management problem in distinct working groups, sometimes lacking common approaches to addressing similar problems, or even different naming to related parameters, raising problems of interpretation and functional logic of the created models.

In addition to that, an overarching, global view on the usage of models does not exist, thus making difficult the definition of common management procedures, use cases and best practices.

All these limitations become stumbling points that require new directions and an opportunity for fostering the development of new control and management capabilities towards autonomous, data-driven networks.

As a final remark on the topic of autonomous networks, where the intention is that either internal or external applications could directly interact with the network, some functional capabilities are yet missing. All the modelling capabilities before are mostly conceived for allowing control and management procedures at the level of service, network or device [RFC8309], where the interaction at service or network level is performed against an SDN controller, while the interaction at device level is done by the SDN controller directly towards the network element.

There is however lack of functional entities for the purpose of exposing network capabilities, possibly integrating additional relevant information that could be required for those applications for taking decisions on the network. Some initial approaches have been proposed [draft-contreras-alto-ietf-nef], but there is not a consolidated view on this yet in IETF.