NMOP
Internet-Draft
Intended status: Informational
Expires: 4 September 2025

M. Mackey
B. Claise
Huawei
T. Graf
Swisscom
H. Keller
Deutsche Telekom
D. Voyer
Bell Canada
P. Lucente
NTT
I. D. Martinez-Casanueva
Telefonica

3 March 2025

Knowledge Graph Framework for Network Operations
draft-mackey-nmop-kg-for-netops-02

Abstract

This document describes some of the problems in modern operations and management systems and how knowledge graphs and RDF can be used to solve closed loop system, in an automatic way.

Discussion Venues

This note is to be removed before publishing as an RFC.

Source for this draft and an issue tracker can be found at https://github.com/mike-mackey.

Status of This Memo

Table of Contents

1. Introduction

The IAB organized a workshop in June 2002 to establish a dialog between network operators and protocol developers, to guide IETF when working on network management protocols and data models. The outcome of that workshop was documented in the "Overview of the 2002 IAB Network Management Workshop" [RFC3535] which identified 14 operator requirements for consideration in future network management protocol design and related data models, along with some recommendations for the IETF.

The RFC3535 requirements were instrumental in developing first the NETCONF protocol (in the NETCONF Working Group) [RFC6241], the associated YANG data modelling language (in the NETMOD Working Group) [RFC7950], RESTCONF [RFC8040], and most recently CORECONF [I-D.ietf-core-comi].

A new IAB workshop, Next Era of Network Management Operations (NEMOPS), is getting organized to tackle the next big challenges in the world of network management. Exactly like the previous workshops, operator challenges and requirements will be documented. The new set of requirements will hopefully guide the Operational and Management Area (OPS) https://datatracker.ietf.org/group/ops/about/ future directions.

This document describes the challenges in network operations, and proposes a new framework based on knowledge graph, to solve (some of) those operational challenges, mainly how to automatically assure networks.

As an introduction, let's review the difference between information model and data model. Quoting RFC 3444 [RFC3444], "The main purpose of an information model is to model managed objects at a conceptual level, independent of any specific implementations or protocols used to transport the data. The degree of specificity (or detail) of the abstractions defined in the information model depends on the modelling needs of its designers. In order to make the overall design as clear as possible, an information model should hide all protocol and implementation details. Another important characteristic of an information model is that it defines relationships between managed objects."

An information model, typically expressed in a language such as Unified modelling Language (UML) do not generate the full APIs, as it lacks some of the implementation- and protocol-specific details; for example, rules that explain how to map managed objects onto lower-level protocol constructs.

A data model, on the other end, can directly be used for network automation. As an example, YANG data models [RFC7950] can generate APIs, to be accessed by protocols such as NETCONF and RESTCONF.

2. Challenges

This section covers the current operational challenges.

2.1. Data Overload from Network Operations

Modern network operators are inundated with vast amounts of data generated from various sources within the network. This data encompasses information from the management plane, control plane, and data plane. Each contributing to a massive influx of information. The sheer volume of data is staggering, making it challenging even for advanced computer systems to process and analyze effectively.

2.2. Difficulties in Data Analysis and Insight Extraction

Data analysts with network domain knowledge play a crucial role in leveraging this data to predict faults, perform Root Cause Analysis (RCA), and implement automatic remediation. However, they often struggle to extract useful information due to the overwhelming volume, data standardization and complexity of the data. The challenge lies not only in processing the data but also in finding meaningful patterns and correlations that can derive actionable insights.

2.3. Complex Data Correlation Requirements

A significant challenge in modern network operations is the need to correlate data from different planes - management, control, and data. Each plane generates its own set of data, often stored in disparate repositories and formats. Linking this information together is essential to gain a holistic view of network operations but is inherently difficult.

2.4. Service and Customer Correlation

Ultimately, all collected data must be correlated back to specific connectivity services (and its customers). This correlation is critical for understanding the impact of network events on service quality and customer experience. However, the process of linking management plane data with control plane and data plane information, is to provide a coherent connectivity service with customer perspective is extremely challenging. Even as a simpler challenge, it's not always easy to correlate the configuration management plane information with the streamed operational data: YANG, as a data modelling
language simplifies the situation, if used for both config and streaming but some extra correlation might anyway be required beyond the data model.

2.5. Data Storage and Format Disparities

Data is frequently stored in multiple repositories, each potentially using different formats. This fragmentation makes it difficult to manage the links between different data sets. In some cases, these links, relationships, may be lost within the engines of the management and analytics systems, leading to incomplete or incorrect analyses.

For example, data is stored using a variety of data model languages, sometimes different schemas for a specific data model language (for example YANG), which are sometimes different per router vendors, often siloed in different applications and storage platforms. Establishing relationships between this data, especially when disconnected from the service context and original intent, is exceedingly difficult. This fragmentation hinders the ability to maintain a cohesive understanding of network operations and their impacts on service quality.

2.6. Contextual Understanding and Relationship Mapping

To reduce the problem space and facilitate automated decision-making, it is essential to understand the context and semantic of the data, the relationships between different data sets, and how this data relates to the overall network. By developing a clear understanding of these relationships, network operators can make more informed and quicker decisions, which is crucial for achieving autonomous operations.

2.7. Loss of Context in Data Collection

The context of the collected data is often lost, complicating the task of network monitoring and analysis. For instance, it can be challenging to determine what a particular interface represents (in other words, its role): is it a Provider Edge (PE), Provider (P), Customer Edge (CE), or an interface on an Autonomous System Boundary Router (ASBR) (or ABR)? Based on this particular context, the intent is obviously different, and, as a consequence, this context is key to interpret the collected data. For example, the IP addresses observed on CE router, PE router, or P router have different context (and on the PE router, it actually depends if we refer to CE-facing or PE-facing interface). As a different example, understanding whether a link serves as a primary connection for certain customers or a backup for others is critical but frequently ambiguous.

2.8. Data Collection Methods and Interpretation

The methods and intervals at which data is collected also vary, which contributes in adding another layer of complexity. Data might be sampled within specific time windows, on-change, on-demand, or periodically. It might represent an aggregation or calculation of other values. Understanding how the router or analytics engine computes this data is vital for accurate analysis and troubleshooting.

2.9. Organizational Silos

In many organizations, network configuration and operations teams function as separate entities. This division can lead to a disconnect where the rationale behind network changes is lost or miscommunicated between teams. This lack of cohesion can further complicate data correlation and analysis efforts.

2.10. Multiple Sources of Truths

What is the network intent? Is it owned in the controller, or the current network is the network intent? What if the network is configured at the same time from the controller and from the CLI (for quick network anomaly resolution), does it imply that the network intent is partially in the controller, and partially in the network state? There are actually multiple sources of truth in networking:

* the controller configuration (intended state)

* the network itself (applied state, described by the Digital Map)

* the inventory, typically stored across different systems, with a different ID (sometimes UUID [RFC7950])

* the IP Address Management (IPAM) is another other source of truth

* etc.

2.11. Machine Readable Knowledge

While we mentioned multiple data sources, with different data modelling languages, the requirement is to have one data sources in a machine readable way, with the ability to correlate and link information.
Note that, sometimes, the modelling language is simply not existent, as protocols such as BMP or BGP-LS directly stream PDUs.

3. IETF Initiatives

To help with the different challenges mentioned in the previous section, the IETF standardized some RFCs, while some IETF drafts are currently being worked on:

TO DO: once we have the exact challenge tags, we are going to refer to those

* Service Assurance for Intent Based Networking [RFC9417] [RFC9418]

* Network Telemetry framework [RFC9232] explains the different telemetry mechanisms

* draft-ietf-nmop-yang-message-broker-integration-03 (https://datatracker.ietf.org/doc/draft-ietf-nmop-yang-message-broker-integration/)
specifies an Architecture for YANG-Push to Message Broker Integration is helping with the data collection aspects.

* draft-ietf-opsawg-collected-data-manifest (https://datatracker.ietf.org/doc/draft-ietf-opsawg-collected-data-manifest/) documents the metadata that ensure that the streaming collected data can be interpreted correctly.

* draft-ietf-nmop-network-anomaly-architecture (https://datatracker.ietf.org/doc/draft-ietf-nmop-network-anomaly-architecture/) defines an architecture for detecting anomalies in the network

* The basic concepts of the Digital Map are mentioned in draft-havel-nmop-digital-map-concept (https://datatracker.ietf.org/doc/draft-havel-nmop-digital-map-concept/)

These are building blocks, to help towards the goals of autonomous networking. However, these building blocks are not sufficient.

4. The Difficult and Costly Data Models Integration with Different Silos Protocol & Data Models

4.1. Understanding And Using Different Models In A Solution

Even excluding the vast amount of vendor specific models, the telecommunication industry is drowning in models from many different SDOs (IETF, TMF, ETSI, ONF, MEF, 3GPP). All of them fulfill a need and all of them are optional depending on the requirements of the solution/operator. Some of the models are designed to be extended, therefore even though a model is based on a standard it can deviate or add new information. This model and API soup results in confusion and in some cases (mis)interpretation that can differ per implementation.

There have been attempts to address this, to converge models and APIs towards a single model. These initiatives have largely failed. There are many reasons for this (technical debt, separation of concerns/responsibility between SDOs) but the reality is that this remains (and will likely always remain) unachievable. So what is needed is a way to understand how these different models connect and how these models were interpreted by the solution designer.

4.2. Example: Onboarding A New Device

In order to onboard a new device, what features and models that device supports must be known, how that device will map to any existing internal models for resource management e.g. https://github.com/Open-Network-Models-and-Interfaces-ONMI/TAPI, Assurance is mapped to existing assurance and collection models (data collection/message broker/TSDBs), existing health assurance pipelines (as described in [draft-ietf-nmop-network-anomaly-architecture]).

If I onboard a new device will it work with my data processing pipeline If I receive observe a problem in my service, can I trace quickly to find the related network configuration, if I decide I need to modify that network configuration do I know the values that must change at the resource API in order for it to happen.

The cost of onboarding a new device or indeed upgrading to a new software/ hardware version is buried within the different applications, the mapping code that transforms data between models, the impact on existing models (is a new instance enough or does the application schema need to be extended)

What if we could answer all of those questions by querying the connections between schemas and data. What if we could trace the connections across different applications and different design artefacts.

4.3. Different Models For Different Jobs

On the other side, with different technology domains and different protocols, come different data models. In order to assure cross domain use cases, the network management system and network operators must integrate all the technologies, protocols, and therefore data models as well. In other words, it must perform the difficult and time-consuming job of integrating & mapping information from different data models. Indeed, in some situations, there exist different ways to model the same type of information.

This problem is compounded by a large, disparate set of data sources: * MIB modules [RFC3418] for monitoring, * YANG models [RFC7950] for configuration and monitoring, * IPFIX information elements [RFC7011] for flow information, * syslog plain text [RFC3164] for fault management, * TACACS+ [RFC8907] or RADIUS [RFC2865] in the AAA (Authorization, Authentication, Accounting) world, * BGP FlowSpec [RFC5575] for BGP filter, * BMP - BGP Monitoring protocol [RFC7854] * BPG-LS for IGP monitoring * etc. or even simply the router CLI for router management.

Some networking operators still manage the configuration with CLI while they monitor the operational states with SNMP/MIBs. How difficult is that in terms of correlating information? Some others moved to NETCONF/YANG for configuration but still need to transition from SNMP/MIBs to Model-driven Telemetry.

When network operators deal with multiple data models, the task of mapping the different models is time-consuming, hence expensive, and difficult to automate.

4.4. Example: Whats An Interface ?

To make it crystal clear, let’s illustrate this with a very simple and well known networking concept: a simple interface. Let’s start with a simple CLI command: "show ip interface" for basic interface information.

* Between MIB module and YANG model, fortunately, we have the same ifIndex concept (ifIndex in MIB and if-index in YANG). This facilitates the mapping.

* In the context of IPFIX models, even if the ingressInterface and egressInterface report the famous ifIndex values within the flow record, the interface semantic changed. We have one specific field for the ingress and another one for egress traffic. The MIB object ifIndex doesn't make that distinction. While it's not difficult to map the IPFIX interface information elements with the MIB and YANG interface ones, the different semantic must be hardcoded in the NMS.

* For the protocols from the AAA world, TACACS+ and RADIUS interfaces are called "ports" to use the right terminology. Those have nothing to do with the networking ifIndex definitions, even if it's perfectly fine to host TACACS+ or RADIUS in routers.

* How to map with interface concept with the syslog message, where the syslog message might not have the exact same interface id (Gigabit Ethernet versus gigE versus gigEthernet ... X/Y.Z as an example) for a machine to read.

At this point, these concepts are known inside network engineer's heads, their network domain knowledge, but how to convey this information to a data scientist lacking the network domain knowledge but capable to analyze data systematically? The cost of documenting this information for the different ways it can be configured/used is enormous. Ideally protocol designers should understand that there will be an network management/automation cross domain use case that will require the integration and the potentially mapping of those different data models.

4.5. How To Connect Information For Closed Loop

Going one step further, understanding that an anomaly defined in [I-D.netana-nmop-network-anomaly-lifecycle] is connected to a symptom created by SAIN [RFC9418], which has an alert that defines an expression based on a set of metrics that are IETF but also vendor defined. The Service(s) that are experiencing an anomaly defined using a Service Intent that was defined using /[TMF921A/] but full filled using [RFC9182] that maps to one or more vendor models.

In order to be able to automate any closed loop action, the relationship between the Service Intent (defined using /[TMF921A/]), the error/policy condition that caused the symptom and the fulfillment provided at the device level via [RFC9182] all have to be understood.

Many of the operators (at the time of writing) who are trying to document and manage these relationships are trying to do it using various spreadsheets and version control systems. Instead, we could provide a way to define and query those relationships in a manner that could instantly answer all the questions that an operator has.

What if we could provide this information not only in a format the operator can understand but in a way a machine can easily interpret and make decisions.

This is the key to moving from Automation to Autonomy and one of the keys to unlocking autonomy is to leverage Knowledge Engineering and Knowledge Based Systems (KBS)

4.6. The Limits of YANG as THE Model Language

While YANG is deployed data model for configuration and monitoring, YANG has limitations that prevent it to be THE model to which other data models will be mapped. Instead, the different data models will still have a life of their own (ex: the IPFIX model does a good job for its purpose). Therefore let use introduce knowledge graph concepts, which will address the challenges.

TO BE COMPLETED.

5. Knowledge Graph Framework

Addressing these challenges mentioned in section 2 requires innovative approaches that can handle the scale and complexity of the data while ensuring accurate correlation and analysis. By leveraging advanced data management techniques and semantic technologies, network operators can unlock the full potential of their data, paving the way for more efficient and autonomous network operations. Understanding the context and relationships of the data collected is essential to overcoming the limitations of traditional siloed approaches and achieving seamless, automated network management.

A knowledge-based system (KBS) is a computer program that leverages a knowledge base and an inference engine to solve complex problems. These systems are designed to simulate human decision-making and problem-solving capabilities, making them valuable tools in various domains. Here are the key features of a knowledge-based system:

5.1. Knowledge Base

The knowledge base is the core component of a KBS, where all the explicit knowledge about the domain is stored. This knowledge is usually represented in a structured and formalized manner to facilitate easy access and manipulation. Key characteristics of the knowledge base include:

* *Explicit Representation*: Knowledge is represented in a way that can be easily interpreted and processed by the system.

* *Structured Data*: Information is organized in a structured format, such as rules, facts, and relationships.

* *Rich Semantics*: The knowledge base captures not only raw data but also the meaning and context of that data.

5.2. Inference Engine

The inference engine is the reasoning component of a KBS. It processes the information stored in the knowledge base to derive new knowledge, make decisions, or solve problems. Key functions of the inference engine include:

* *Reasoning*: Applies logical rules to the knowledge base to infer new information or conclusions.

* *Decision-Making*: Uses the inferred knowledge to make decisions or recommend actions.

* *Problem Solving*: Solves complex problems by systematically exploring possible solutions based on the knowledge base.

  In some proposed architectures the inference engine is split into individual agents that have responsability for a decomposed aspect of the Service/ Network lifecycle (e.g. anomaly detection, assurance remediation, solution proposal, solutions evaluation, solution actuation etc). The Agents (which could be AI Agents) can communicate/collaborate via the Knowledge Base.

5.3. Formal Ontology

A formal ontology is a crucial element in capturing facts about the world in which the KBS operates. It provides a structured and formal description of the concepts and relationships within a specific domain. Key aspects of formal ontologies include:

* *Concepts and Relationships*: Defines the key concepts and the relationships between them within a particular domain.

* *Standardized Vocabulary*: Provides a common vocabulary for the domain, ensuring consistency and interoperability.

* *Formal Specification*: Uses formal logic to specify the properties and constraints of the concepts and relationships, allowing for precise and unambiguous interpretation.

5.4. Comprehensive and Dynamic Knowledge

Knowledge-based systems are designed to handle a comprehensive range of knowledge within their domain and adapt to new information. This includes:

* *Extensibility*: The ability to add new knowledge and rules to the system as the domain evolves.

* *Adaptability*: The capability to update and refine the knowledge base based on new insights or changes in the environment.

* *Integration*: Combining knowledge from multiple sources to provide a holistic understanding of the domain.

6. FAIR data

FAIR Data Principles were defined in a 2016 research paper by a consortium of scientists and organizations in Nature.
The authors intended to provide guidelines to improve the Findability, Accessibility, Interoperability, and Reuse of digital assets. They have since published their principles in https://www.go-fair.org/fair-principles/.

6.1. Findability (F)

Networking systems generate large amounts of configuration, telemetry, and state data, which needs to be easily discoverable by network operators, engineers, or automated systems.

6.2. Accessibility (A)

Data within the networking domain needs to be accessible to both human operators and automated systems, with well-defined access mechanisms and protocols.

6.3. Interoperability (I)

Networking environments typically involve many devices and protocols from different vendors. Ensuring interoperability across systems is crucial for seamless network operations and management.

6.4. Reusability (R)

Reusability ensures that network data can be used and repurposed across different contexts, applications, and scenarios.

The FAIR principles have been widely cited, endorsed and adopted by a broad range of stakeholders since their publication in 2016. By intention, the 15 FAIR guiding principles do not dictate specific technological implementations, but provide guidance for improving Findability, Accessibility, Interoperability and Reusability of digital resources.

6.5. Creating And Using FAIR Knowledge Graphs

There are two major approaches to implementing knowledge graphs. Property graphs and RDF. In recent years Property graphs have gained a lot of traction with successful with companies like Neo4j and others attempting to standardize on an approach.

Property graphs are great to use in a closed application but face a number of issues when moving to large scale and open data that are designed to be FAIR. For example:

* *Schema*: Property Graphs do not have a schema. This can be considered a positive as well as negative, but having a schema that you can validate against can limit issues and bugs during implementations

* *Validation*: Property graphs do not define a way to validate data but the W3C standard, SHACL (SHApes Constraint Language) to specify constraints in a model driven fashion.

* *Globally Unique Identifiers*: The identifiers in Property Graphs are strictly local. They donâ\200\231t mean anything outside the context of the immediate database.

* *Resolvable Identifiers*: Because URI/IRIs are so similar to URLs, and indeed in many situations are URLs it makes it easy to resolve any item in RDF graph.

* *Federation*: While there are proprietary mechanisms for federating property graphs across databases e.g. neo4j fabric, federation is built into SPARQL the W3C standard for querying â\200\234triple storesâ\200\235 or RDF based Graph Databases.

There are ways for these two worlds to converge though, there is current work within the w3c to add properties to edges in RDF. This work RDF-star (https://w3c.github.io/rdf-star/cg-spec/editors_draft.html) (RDF-_) and SPARQL-star (SPARQL-_) at time of writing is ongoing in the W3C.

Similarly, Neo4j has a plugin "Neosemantics" that enables the use of RDF data and some of the RDF stack (OWL,RDFS,SHACL) inside of Neo4j but crucially using Cipher and not SPARQL for querys.

So as of now, RDF Knowledge Graphs have FAIR baked in and are part of the standard. Property graph approaches have proprietary solutions to help make things FAIR but there is no standard. These two worlds and approaches do seem to be converging though.

7. Introduction to the Semantic Web Technology Stack

The Semantic Web technology stack enables more sophisticated data integration, sharing, and retrieval across diverse information systems. At its core, it provides a framework for defining, linking, and querying data on the web, allowing for more intelligent and interconnected systems. Here is an overview of the key components of the Semantic Web technology stack:

7.1. URI/IRI: Uniform Resource Identifier/Internationalized Resource Identifier

A Uniform Resource Identifier (URI) is a unique sequence of characters that identifies a logical or physical resource used by web technologies. URIs may be used to identify anything, including real-world objects such as people and places, concepts, or information resources like web pages and books. The Internationalized Resource Identifier (IRI) extends the URI to include a wider range of characters from different languages, facilitating global use and interoperability.

7.2. RDF: Resource Description Framework

The Resource Description Framework (RDF) allows you to link resources (concepts) together in a way that forms a directed graph. For example, you could represent the statement "John is a person" using RDF. However, RDF alone does not provide the means to classify objects or establish complex relationships such as saying that a person is a subclass of human beings.

7.3. RDFS: RDF Schema

RDF Schema (RDFS) builds upon RDF by providing more expressive vocabulary to classify resources and establish hierarchical relationships. Using RDFS, you can define classes and subclasses (using rdfs:class and rdfs:subclass), and set restrictions on properties (relationships) within your domain knowledge using rdfs:domain and rdfs:range. This allows for a more structured and meaningful representation of data.

7.4. OWL: Web Ontology Language

The Web Ontology Language (OWL) enhances the expressiveness of RDFS by introducing more detailed and complex constraints on data. OWL categorizes properties into object properties (relationships between two resources) and data properties (relationships between a resource and a data value). It also allows you to add restrictions on properties, such as specifying cardinality constraints or defining equivalent and disjoint classes, enabling more precise and sophisticated knowledge representation.

7.5. Query: SPARQL Protocol and RDF Query Language

SPARQL (SPARQL Protocol and RDF Query Language) is an RDF query language designed for querying and manipulating data stored in RDF format. SPARQL is powerful because it can automatically join all objects in a graph from a single query, allowing for complex and efficient data retrieval. This capability makes SPARQL an essential tool for accessing and integrating diverse datasets within the Semantic Web framework.

Semantic Web technologies have significantly evolved beyond their initial web-based applications, extending into various industries (Healthcare, Finance, Manufacturing, Government and Public Sector) as a powerful means to define, integrate, and retrieve knowledge.

7.6. Validation: Shapes Constraint Language (SHACL)

SHACL (Shapes Constraint Language) can be used to enhance an RDF-based solution by providing a formal mechanism for validating the structure, content, and constraints of RDF data that models network devices, configurations, and relationships.

By using SHACL, you can ensure that the data adheres to predefined business rules, network policies, and industry standards, thus improving data quality and consistency within a management system.

7.6.1. Ensuring Data Consistency

For instance, you can use SHACL to enforce that each network device has a deviceType (e.g., router, switch) and an associated IP address, and that routers have specific attributes such as a bgpAsn (BGP Autonomous System Number).

7.6.2. Validating Relationships

For relationships between objects, SHACL allows you to validate these interconnections by specifying shapes that ensure the correct relationships are maintained.

7.6.3. Enforcing Network Policies

In network management, policies can dictate how devices are configured and how they interact. For example, a policy may require that certain VLANs are used in specific types of networks or that certain subnets are restricted to specific devices.

SHACL allows you to encode these policies as constraints and automatically validate the RDF data against them. For instance, if a policy requires that only certain VLANs are allowed on specific switches, you can define a SHACL shape to validate this.

7.6.4. Automating Configuration Compliance

In large-scale networks, automating compliance checks is essential to ensure devices are configured according to standards and policies. SHACL can be used to automatically validate the entire RDF dataset representing network configurations against predefined shapes. This can flag potential issues such as missing configurations, incorrect relationships, or policy violations, enabling network administrators to quickly take corrective action.

7.6.5. Error Reporting and Diagnostics

SHACL provides detailed error reporting and diagnostics, which can help network administrators quickly identify and fix issues in the network configuration. For each violation of a SHACL shape, SHACL can generate meaningful error messages, such as missing required properties, invalid data types, or relationships that do not conform to the network topology.

8. Why Semantic Web is Right for the Networking World?

8.1. Handling Vast Amounts of Data

Modern network operators collect extensive data from various network planes - Management, Control, and Data. Semantic Web technologies are designed to handle large datasets efficiently:

* *RDF (Resource Description Framework)* enables the modelling of data as a directed graph, allowing for flexible and scalable data representation.

* *SPARQL* can efficiently query large datasets, automatically joining relevant data points to provide comprehensive insights.

* *URI/IRI* allow data to be referenced and enriched using markup/ metadata without modifying the existing systems. Information can referenced and retrieved using the schema/metadata defined in RDF.

8.2. Improved Data Correlation and Integration

Semantic Web technologies excel at linking disparate data sources and formats, which is crucial for network operations:

* *URI/IRI* provides a unique way to identify and link resources across different data silos, ensuring that all data points can be correlated accurately.

* *RDFS (RDF Schema)* and *OWL (Web Ontology Language)* provide mechanisms to classify and relate data, enabling the creation of detailed ontologies that represent network components and their relationships.

8.3. Contextual Understanding and Enhanced Metadata

One of the major challenges is the loss of context in the data collected from network operations:

* *RDFS* and *OWL* allow operators to define rich metadata about network elements. For instance, they can specify that an interface is a Provider Edge (PE) or Customer Edge (CE), and whether a link is primary or backup.

* *OWL* provides advanced features to define and enforce data properties and relationships, ensuring that the context of data is maintained and understood correctly.

8.4. Data Interoperability Across Multiple Repositories

Network data often resides in multiple repositories with different schemas and formats:

* *RDF* provides a common framework for data representation, enabling interoperability between diverse data sources.

* *Linked Data principles* can connect data from various repositories, making it easier to integrate and query across different systems.

* *Consume Or Reference Data From Multiple Sources in Multiple Formats* using well known patterns within the semantic web ecosystem for connecting external databases, whether relational, hierarchical, tabular or other graph formats.

* *Deterministic URIs* can allow data can be referenced remotely without being consumed or replicated inside a knowledge store. Thus allowing only data that is enriched to be created and stored in the knowledge and for it to reference the existing data in externally repositories.

8.5. Enhanced Fault Prediction and Automated Remediation

To predict faults and automate remediation, operators need to link and analyze data from different network planes:

* *SPARQL* can query across multiple datasets, linking management, control, and data plane information to provide a holistic view of the network.

* *OWL & RDF(S)* can define rules, relationships and constraints that breakdown the barriers between the network planes and link this data with all relevant information (e.g. context based on topologies represented by [I-D.havel-nmop-digital-map], configuration based on network element YANG).

8.6. Bridging Organizational Silos

Network configuration and operations teams often work in silos, leading to miscommunication and data fragmentation:

* *Ontologies* defined using *RDFS* and *OWL* can standardize the terminology and relationships used across teams, ensuring consistent understanding and communication.

* *Semantic annotations* can capture the intent and rationale behind network changes, preserving context and facilitating better collaboration. Importing existing schemas (whether from YANG or Relational or something else) and defining the connections between them allow the semantic of any object or value to fully known and traced within the system.

8.7. Managing Schema and Format Disparities

Different data formats (YANG, IPFIX, BMP) and schemas can make data integration challenging:

* *Semantic Web technologies* provide a unified model to represent data from various formats, enabling seamless integration and retrieval.

* *Schema mapping* using *RDF* and *OWL* can reconcile differences between schemas, providing a coherent view of the data.

It’s not only about protocol and models (IETF), we can link to the NMS/OSS layers, from the top down to the bottom up ... but also (business) intent, the BSS.

9. YANG and RDF

As mentioned above, there are more than enough models already in the telecom domain. Chief among them (from the IETF point of view) is YANG. The YANG modelling language already has many ways to augment and extend the model, but these extensions are very formal and not very dynamic.

9.1. Data catalog for YANG data sources

The flexibility and extensibility of knowledge graphs have made them a popular choice for implementing data catalogs. The purpose of a data catalog is to provide consumers with a registry of datasets exposed by data sources where to find data of interest. Additionally, these datasets can be linked to the (business) concepts that they refer to, so that consumers can search for datasets based

on relevant concepts such as â\200\234interfaceâ\200\235.

Knowledge graphs can enable the YANG Catalog to evolve towards a data catalog, where the YANG modules represent datasets of interest. The dependencies between YANG models (import, deviations, augments) can be naturally represented in the knowledge graph. In turn, these YANG models can be linked with concepts that are represented in ontologies.

Additionally, these YANG models, can be combined with the implementation details of network devices yang lib augment [I-D.lincla-netconf-yang-library-augmentation] that could be part of an inventory [I-D.ietf-ivy-network-inventory-yang].

9.2. Translation of YANG to RDF

Since the original YANG specification [RFC6020], IETF has embraced YANG as the way to define any new models or APIs, from the device all the way up the Service model [RFC8299]. How easy would it be to convert these vast model set to RDF ?.

RDF has its roots in semantic web and is defined by the w3c, who are also the owners of the XML standard. The original [RFC6020] definition for YANG has XML deeply embedded, defining serialization formats for the YANG (YIN) in XML as well as of course instances of YANG data used by NETCONF.

Translation of XML to RDF is a well described problem and many tools exist in order to achieve it, w3c themselves have R2RML that allow custom definitions of mappings.

Generation of IRIs for YANG schema objects can be created using schema paths and IRIs for YANG instance data using XPaths. There are several advantages to this approach, the most important being that the IRI is deterministic and could be generated externally by a knowledge system without need to access the real data. There is a second advantage that it is human readable and therefore easy to browse.

The main disadvantage being the possible length of IRIs and the volume of data being processed could be lead to memory/performance issues. There are obvious trade offs to be explored but it can be seen that YANG is very much a good fit for modelling as knowledge in RDF give both formats close association to XML.

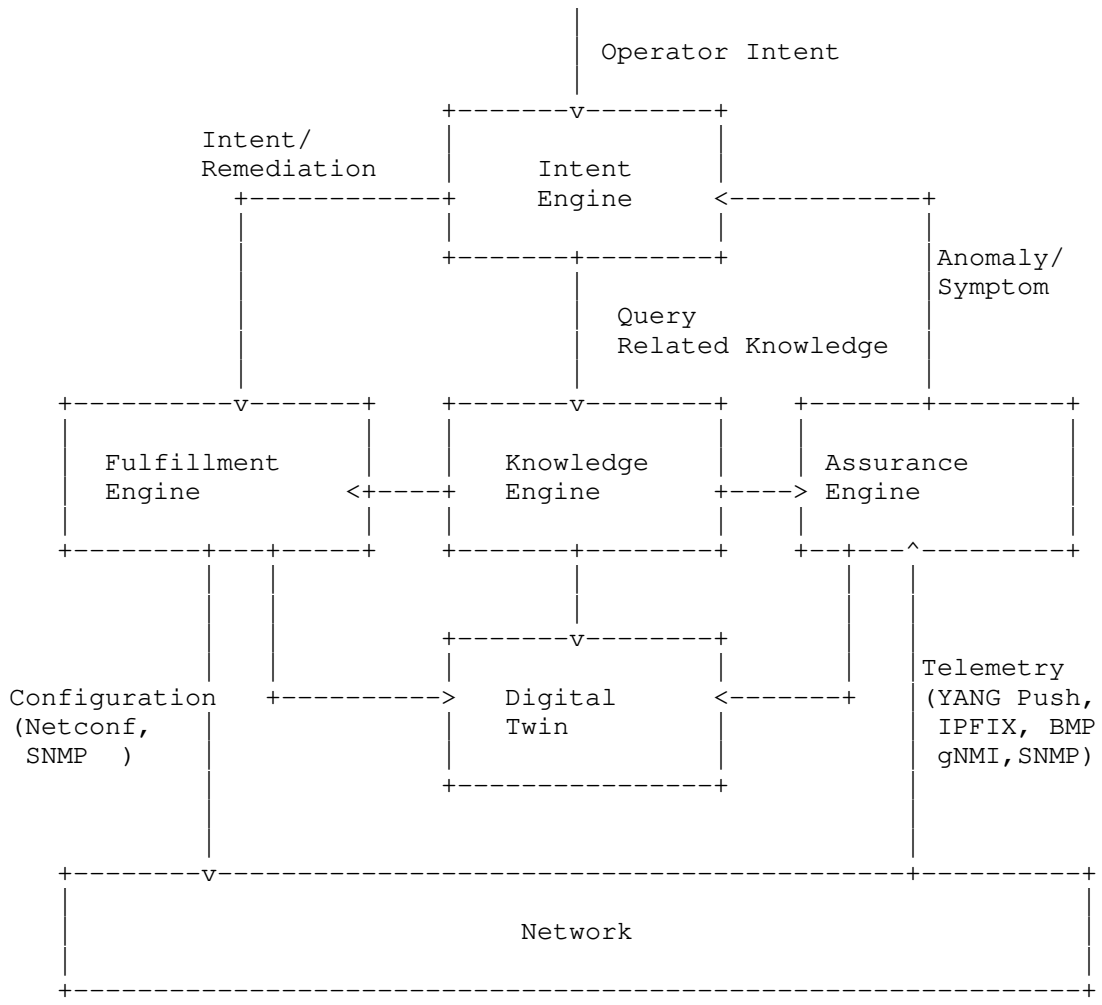
10. Knowledge Engine Positioning And Architecture

Below shows the basic positioning of the Knowledge Engine in any OSS system. As you can see the Knowledge Engine is at the heart of any decision making process.

Key to this is the ability to consume and connect data from multiple different datasources and to connect them in a single semantic layer.

Note as mentioned above, there are already many models that exist in telecommunication systems, the goal maybe not to create a new model but to provide a simple way to navigate between the existing models. Understanding that the value on the intent API that is mapped to a value on the fulfillment API that is used to configure this part of the device is connected to the Telemetry metric that was received where an anomaly was observed.

This 360 degree view of the network is the only way the secrets of the network can be unlocked and autonomous networks enabled.



10.1. Key Use Cases For Knowledge Engine

The above shows the target for Autonomous networks and automated decision processing, but for each step the Knowledge Engine can play a key role.

10.1.1. Service Intent Translation

A knowledge graph can facilitate intent translation the operator intent to the network intent by providing a unified way to query the digital twin [I-D.irtf-nmrg-network-digital-twin-arch].The ability to integrate heterogenous silos of data, in combination with the explicit representation of the semantics of the data; making the knowledge graph a powerful technology for building and connecting data across different datasource.

The capability to represent abstract concepts by means of ontologies, enables the representations of a generic network digital twins, regardless of the complexities of the underlying technologies. For example, an abstract representation of a network topology Digital Map [I-D.havel-nmop-digital-map] in the knowledge graph can be translated into a descriptor or data model that is specific to the technology used.

10.1.2. Contextualized telemetry data

Having context of how YANG telemetry data [I-D.ietf-opsawg-collected-data-manifest] is being collected can improve the understanding of the data for network analytics or closed-loop automation. Knowledge graphs can help in this task by linking the collected data with**: (i) the metadata that characterizes the platform producing the data; and (ii) the metadata that characterizes how and when the data were metered.

10.1.3. Anomaly detection and incident management

Knowledge graphs can help in the detection of anomalies in network systems by linking event/metric data (e.g. logs, alarms, and ticketing) with the context (digitial twin/network configuration). The Knowledge graph enables the connecting of these events using the context to link and explore for new connections.

10.1.4. Network Rectification:

Combing all of the above to enable the OSS to respond to issues in the network and automatically generate a change in the network to rectify the problem.

10.2. Accessing Existing Data

A key enabler that allows the information in all these systems to be exposed and connected is the Virtual Knowledge Graph. Originally called Ontology Based Data Access (OBDA), it is a collection of techniques and technologies that help overcome the challenge of combining data from different sources and formats. It uses ontologies to create a unified view of this data, and mappings to link the ontology with the individual data sources.

OBDA has evolved through three main stages:

* Materialization: Initially, OBDA focused on translating data into a common format and storing it in a central location, similar to data warehousing.

* Query Translation: To avoid the limitations of materialization, OBDA shifted towards translating queries over the unified view into queries specific to each data source.

* Declarative Mappings: The latest generation of OBDA uses declarative mapping languages, like R2RML, to define how data sources relate to the ontology. This improves flexibility and simplifies the integration process.

Virtual Knowledge Graphs provide significant advantages for data integration:

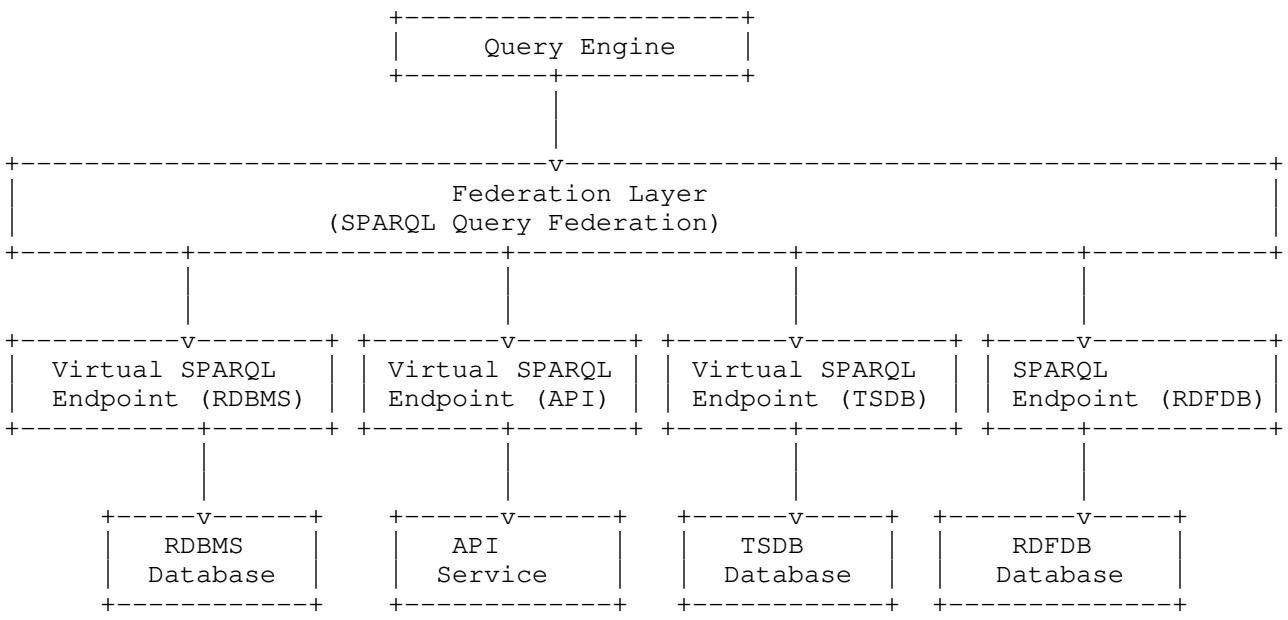
* Real-time Data Access: Data remains in its original sources, ensuring that queries always reflect the latest updates.

* Reduced Costs: Avoid the expense of building and maintaining a separate, materialized data store.

* Simplified Integration: Leverages your existing data infrastructure and expertise.

* Increased Agility: Supports an incremental approach to integration, making it easier to adapt to changes and add new data sources over time.

Unlike traditional materialized approaches that require ETL to create a unified data copy, Virtual Knowledge Graphs offer a more dynamic solution. Instead of moving data, Virtual KGs leave data in place and access it on demand. This eliminates data duplication, reduces latency, and simplifies maintenance, making it a powerful alternative for modern data integration needs.



In the Virtual Knowledge Graph (or Ontology Based Data Access) the remote schema can be imported as RDF/OWL data and used for both query and for creating new relationships over existing data. In this way existing models can be imported and the connections between silos can overlaid as extra knowledge. These relationships can be created manually or programmatically.

At query time, these relationships can be exploited to join data from different sources, allowing the connection between anomaly to assurance metric to inventory object to digital map to configuration to be traced seamlessly regardless of where the information is being stored.

10.3. What is materialised in RDF and what is Virtual ?

Given the above, you may ask what is materialised in the triple store and what is virtual, of course the answer as always is, it depends. If you have a read API that lets you access the remote data anyway you want it e.g. JDBC then maybe virtual is good enough. If however you have a restricted API that makes it difficult to query and join data (e.g. Netconf/Restconf) you may want to import that data into the graph in order to satisfy all of your queries. For this reason we have focused the initial implementation on materializing YANG schema and YANG Instance data.

11. Implementation Status

At IETF Hackathon 121 (Dublin) we successfully demonstrated an approach for translation of YANG schema models to a representation in RDF. This code is available here: https://github.com/Huawei-IOAM/yang2rdf.

At IETF Hackathon 122 (Bangkok) we will demonstrate how that YANG RDF Schema data can be used to create RDF versions of configuration data.

12. Some pointers to existing work for linked data

Linked data and semantic web has already been embraced by TMF and ETSI, their reasons for adoption are both valid both for them and for the IETF when aiming to link data in different systems and to capture knowledge.

12.1. NGSI-LD (Next Generation IoT Services Layer - Lightweight Data)

NGSI-LD is an ETSI standardized framework for representing and managing context information in the Internet of Things (IoT) domain.

Context Information Model: Represents context information as entities with properties and relationships, inspired by property graphs. *Semantic Foundation:* Based on RDF (Resource Description Framework) for formal semantics and linked data principles.

12.2. TMF921A Intent Management API

TMF tmf921a is an API for intent based networking. It aims to provide a standardized interface for expressing and managing high-level network intents, enabling automated network configuration and optimization.

Knowledge Based Reasoning: Core to the TMF approach is the ability for the intent management functions to "use reason intrinsically by examining the relationships between facts". Therefore their decision to model the intents as knowledge and to use *RDF* to define the intents.

13. Next Steps for the Industry

It’s important to note, the authors are not proposing creating a new model for the network, there is much work being done in all of the existing SDOs with numerous models managing all aspects of the network and business. The authors are proposing ways to connect all of this data and through those connections find the semantic of the information and allow it to unlock the knowledge already being

generated by the network.

To this end, the industry must come together to define ways to describe the connections between data (either at the instance level or at the schema level). Agree on formats for importing existing protocol schemas into RDF e.g. in IETF: YANG, IPFIX, BMP but also other models in different SDOs

Crucial to this is to define a way to create deterministic IRIs for both model and instance data so that information in disparate systems and repositories can be referenced, linked and enriched using the power of Knowledge graphs and linked data.

14. Security Considerations

TBC

15. IANA Considerations

This document has no actions for IANA.

16. Reference

16.1. Normative References

16.2. Informative References (to be included)

* RDF 1.1 Concepts and Abstract Syntax. W3C Recommendation, 2014.
* RDF Schema 1.1. W3C Recommendation, 2014.
* OWL 2 Web Ontology Language Document Overview. W3C Recommendation, 2012.
* SPARQL 1.1 Query Language. W3C Recommendation, 2013.
* SHACL - Shapes Constraint Language. W3C Recommendation, 2017.
* R2RML - RDB to RDF Mapping Language. W3C Recommendation, 2012.
* RML - Extend R2RML to allow mapping from any data source. v1.1.2, 2024.

17. References

17.1. Normative References

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <https://www.rfc-editor.org/rfc/rfc7950>.

17.2. Informative References

[ANSA] Pedro Martinez-Julia, Ved P. Kafle, Hitoshi Asaeda., "Application of Category Theory to Network Service Fault Detection. IEEE Open Journal of the Communications Society 5 (2024): 4417-4443.", n.d..

[EERVC] Pedro Martinez-Julia, Ved P. Kafle, Hiroaki Harai., "Exploiting External Events for Resource Adaptation in Virtual Computer and Network Systems, IEEE Transactions on Network and Service Management 15 (2018): 555-566.", n.d..

[I-D.havel-nmop-digital-map] Havel, O., Claise, B., de Dios, O. G., Elhassany, A., and T. Graf, "Modeling the Digital Map based on RFC 8345: Sharing Experience and Perspectives", Work in Progress, Internet-Draft, draft-havel-nmop-digital-map-02, 21 October 2024, <https://datatracker.ietf.org/doc/html/draft-havel-nmop-digital-map-02>.

[I-D.ietf-core-comi] Veillette, M., Van der Stok, P., Pelov, A., Bierman, A., and C. Bormann, "CoAP Management Interface (CORECONF)", Work in Progress, Internet-Draft, draft-ietf-core-comi-19, 3 November 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-core-comi-19>.

[I-D.ietf-ivy-network-inventory-yang] Yu, C., Belotti, S., Bouquier, J., Peruzzini, F., and P. Bedard, "A Base YANG Data Model for Network Inventory", Work in Progress, Internet-Draft, draft-ietf-ivy-network-inventory-yang-05, 28 February 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-ivy-network-inventory-yang-05>.

[I-D.ietf-opsawg-collected-data-manifest] Claise, B., Quilbeuf, J., Lopez, D., Martinez-Casanueva, I. D., and T. Graf, "A Data Manifest for Contextualized Telemetry Data", Work in Progress, Internet-Draft, draft-ietf-opsawg-collected-data-manifest-06, 3 March 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-opsawg-collected-data-manifest-06>.

[I-D.irtf-nmrg-network-digital-twin-arch]
Zhou, C., Yang, H., Duan, X., Lopez, D., Pastor, A., Wu, Q., Boucadair, M., and C. Jacquenet, "Network Digital Twin: Concepts and Reference Architecture", Work in Progress, Internet-Draft, draft-irtf-nmrg-network-digital-twin-arch-10, 28 February 2025, <https://datatracker.ietf.org/doc/html/draft-irtf-nmrg-network-digital-twin-arch-10>.

[I-D.lincla-netconf-yang-library-augmentation]
Lin, Z., Claise, B., and I. D. Martinez-Casanueva, "Augmented-by Addition into the IETF-YANG-Library", Work in Progress, Internet-Draft, draft-lincla-netconf-yang-library-augmentation-01, 4 March 2024, <https://datatracker.ietf.org/doc/html/draft-lincla-netconf-yang-library-augmentation-01>.

[I-D.netana-nmop-network-anomaly-lifecycle]
Riccobene, V., Roberto, A., Graf, T., Du, W., and A. H. Feng, "An Experiment: Network Anomaly Lifecycle", Work in Progress, Internet-Draft, draft-netana-nmop-network-anomaly-lifecycle-05, 3 November 2024, <https://datatracker.ietf.org/doc/html/draft-netana-nmop-network-anomaly-lifecycle-05>.

[RFC2865] Rigney, C., Willens, S., Rubens, A., and W. Simpson, "Remote Authentication Dial In User Service (RADIUS)", RFC 2865, DOI 10.17487/RFC2865, June 2000, <https://www.rfc-editor.org/rfc/rfc2865>.

[RFC3164] Lonvick, C., "The BSD Syslog Protocol", RFC 3164, DOI 10.17487/RFC3164, August 2001, <https://www.rfc-editor.org/rfc/rfc3164>.

[RFC3418] Presuhn, R., Ed., "Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3418, DOI 10.17487/RFC3418, December 2002, <https://www.rfc-editor.org/rfc/rfc3418>.

[RFC3444] Pras, A. and J. Schoenwaelder, "On the Difference between Information Models and Data Models", RFC 3444, DOI 10.17487/RFC3444, January 2003, <https://www.rfc-editor.org/rfc/rfc3444>.

[RFC3535] Schoenwaelder, J., "Overview of the 2002 IAB Network Management Workshop", RFC 3535, DOI 10.17487/RFC3535, May 2003, <https://www.rfc-editor.org/rfc/rfc3535>.

[RFC5575] Marques, P., Sheth, N., Raszuk, R., Greene, B., Mauch, J., and D. McPherson, "Dissemination of Flow Specification Rules", RFC 5575, DOI 10.17487/RFC5575, August 2009, <https://www.rfc-editor.org/rfc/rfc5575>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://www.rfc-editor.org/rfc/rfc6020>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/rfc/rfc6241>.

[RFC7011] Claise, B., Ed., Trammell, B., Ed., and P. Aitken, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information", STD 77, RFC 7011, DOI 10.17487/RFC7011, September 2013, <https://www.rfc-editor.org/rfc/rfc7011>.

[RFC7854] Scudder, J., Ed., Fernando, R., and S. Stuart, "BGP Monitoring Protocol (BMP)", RFC 7854, DOI 10.17487/RFC7854, June 2016, <https://www.rfc-editor.org/rfc/rfc7854>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/rfc/rfc8040>.

[RFC8299] Wu, Q., Ed., Litkowski, S., Tomotaki, L., and K. Ogaki, "YANG Data Model for L3VPN Service Delivery", RFC 8299, DOI 10.17487/RFC8299, January 2018, <https://www.rfc-editor.org/rfc/rfc8299>.

[RFC8907] Dahm, T., Ota, A., Medway Gash, D.C., Carrel, D., and L. Grant, "The Terminal Access Controller Access-Control System Plus (TACACS+) Protocol", RFC 8907, DOI 10.17487/RFC8907, September 2020, <https://www.rfc-editor.org/rfc/rfc8907>.

[RFC9182] Barguil, S., Gonzalez de Dios, O., Ed., Boucadair, M., Ed., Munoz, L., and A. Aguado, "A YANG Network Data Model for Layer 3 VPNs", RFC 9182, DOI 10.17487/RFC9182, February 2022, <https://www.rfc-editor.org/rfc/rfc9182>.

[RFC9232] Song, H., Qin, F., Martinez-Julia, P., Ciavaglia, L., and A. Wang, "Network Telemetry Framework", RFC 9232, DOI 10.17487/RFC9232, May 2022, <https://www.rfc-editor.org/rfc/rfc9232>.

[RFC9417] Claise, B., Quilbeuf, J., Lopez, D., Voyer, D., and T. Arumugam, "Service Assurance for Intent-Based Networking Architecture", RFC 9417, DOI 10.17487/RFC9417, July 2023, <https://www.rfc-editor.org/rfc/rfc9417>.

[RFC9418] Claise, B., Quilbeuf, J., Lucente, P., Fasano, P., and T. Arumugam, "A YANG Data Model for Service Assurance", RFC 9418, DOI 10.17487/RFC9418, July 2023, <https://www.rfc-editor.org/rfc/rfc9418>.

[TKDP] Pedro Martinez-Julia, Ved P. Kafle, Hitoshi Asaeda., "Telemetry Knowledge Distributed Processing for Network Digital Twins and Network Resilience. NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium (2023): 1-6.", n.d..

Appendix A. Acknowledgments

The authors would like to thank Peter Cautley and Anatolii Pererva for providing the appendix example. Professor Declan O'Sullivan and Brad Peters for providing review comments.

Appendix B. Appendix

Appendix C. Resource Description Framework (RDF) schema

The RDF Schema defines the different types of relationship (RDF properties). They are defined hierarchically. That allows specific relationships to be grouped as more generalized relationships. Queries can be applied at different levels of specificity.

```
:ipfixRefersTo a rdf:Property ;
    rdfs:comment "IPFIX reference to a leaf" ;
    rdfs:subPropertyOf :refersTo .

:ipfixRefersToInterface a rdf:Property ;
    rdfs:comment "IPFIX reference to a leaf" ;
    rdfs:subPropertyOf :ipfixRefersTo .
```

Appendix D. SPARQL Protocol and RDF Query Language (SPARQL)

SPARQL finds different relationships that exist between data sources e.g. The relationship "ipfixRefersToInterface" (defined in the RDF schema) will find relationships between any IPFIX data field and the Device Interface. A more generalized relationship "ipfixRefersTo" would find all known relationships between IPFIX and Device (underlay, overlay) Configuration.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX yang: <http://localhost/yang#>
SELECT ?source_path ?relationship ?target_path
WHERE {
  ?source ?relationship ?target .
  ?relationship rdfs:subPropertyOf* yang:ipfixRefersToInterface .
  ?source yang:path ?source_path .
  ?target yang:path ?target_path .
}
```

Appendix E. RESULT

The result for the above query shows - the source: IPFIX fields aligned with IANA "IP Flow Information Export (IPFIX) Entities" - ingressInterface element Id 10 - egressInterface element Id 14 - the type of relationship: (specified in the RDF schema) - the target: YANG path specified in Device Configuration

source	relationship	target
"ingressInterface"	<http://localhost/yang#ipfixRefersToInterface>	"ifm/interfaces/interface/index"
"egressInterface"	<http://localhost/yang#ipfixRefersToInterface>	"ifm/interfaces/interface/index"

Appendix F. SHACL

SHACL (Shapes Constraint Language) can be used to enhance an RDF-based solution for network management by providing a formal mechanism for validating the structure, content, and constraints of RDF data that models network devices, configurations, and relationships. By using SHACL, you can ensure that the data adheres to predefined business rules, network policies, and industry standards, thus improving data quality and consistency within a network management system.

Example of SHACL to validate every router that uses the BGP protocol has a valid BGP ASN configured. ~~~~ ex:BGPComplianceShape a sh:NodeShape ; sh:targetClass ex:Router ; sh:property [sh:path

```
   ex:routingProtocol ; sh:hasValue "bgp" ; ] ; sh:property [ sh:path
   ex:bgpAsn ; sh:minCount 1 ; sh:datatype xsd:integer ; sh:message
   "Routers using BGP must have a BGP ASN defined." ; ] ; ~~~~
```

SHACL can also be used to validate relationships between objects, here is an example of a rule that says each router must be connected to at least one switch.

```
ex:RouterSwitchConnectionShape a sh:NodeShape ;
    sh:targetClass ex:Router ;
    sh:property [
        sh:path ex:connectedTo ;
        sh:class ex:Switch ;
        sh:minCount 1 ;
        sh:message "Each router must be connected to at least one switch." ;
    ] ;
```

Authors' Addresses

Michael Mackey
Huawei
Ireland
Email: michael.mackey@huawei.com

Benoit Claise
Huawei
Belgium
Email: benoit.claise@huawei.com

Thomas Graf
Swisscom
Binring 17
CH-8045 Zurich
Switzerland
Email: thomas.graf@swisscom.com

Holger Keller
Deutsche Telekom
Germany
Email: Holger.Keller@telekom.de

Dan Voyer
Bell Canada
Canada

Email: daniel.voyer@bell.ca

Paolo Lucente
NTT
Veemweg 23
3771 Barneveld
Netherlands
Email: paolo@ntt.net

Ignacio Dominguez Martinez-Casanueva
Telefonica
Spain
Email: ignacio.dominguezmartinez@telefonica.com

Network Management Operations
Internet-Draft
Intended status: Informational
Expires: 30 August 2025

I. D. Martinez-Casanueva
L. Cabanillas
Telefonica
P. Martinez-Julia
NICT

26 February 2025

Knowledge Graph Construction from Network Data Sources
draft-marcas-nmop-kg-construct-00

Abstract

This document discusses the mechanisms that support the management and creation of knowledge graphs from data sources specific to the network management domain. The document provides background on core aspects such as ontology development, identifies methodologies and standards, and shares guidelines for integrating network data sources.

About This Document

This note is to be removed before publishing as an RFC.

The latest revision of this draft can be found at https://idomingu.github.io/knowledge-graph-yang/draft-marcas-knowledge-graph-yang.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-marcas-nmop-kg-construct/.

Discussion of this document takes place on the Network Management Operations Working Group mailing list (mailto:nmop@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/nmop/. Subscribe at https://www.ietf.org/mailman/listinfo/nmop/.

Source for this draft and an issue tracker can be found at https://github.com/idomingu/knowledge-graph-yang.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Table of Contents

1. Introduction

Knowledge graph introduces a new paradigm in data management that facilitates the integration of heterogenous data silos thanks to a semantic layer. In the case of network management, knowledge graphs provide a data integration solution that can cope with the diverse network data sources and telemetry mechanisms [I-D.mackey-nmop-kg-for-netops].

The construction of knowledge graphs is a challenging activity that requires the combination of skills in semantic modelling and data engineering. Semantic data models are represented by ontologies and other forms of structured knowledge, which must be kept in sync with the data pipelines that integrate the different data silos into the knowlede graph. The data integration process is based on the ingestion of raw data from their data sources, the mapping of the raw data to the respective ontologies, and the transformation of the data into a graph structure semantically-annotated.

In this sense, Knowledge Graph Construction (KGC) underpinned by two pillars: i) ontology development; and ii) knowledge graph construction pipeline. These pillars are described in detail in the following sections.

2. Conventions and Definitions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2.1. Terminology

This document defines the following terms:

Data integration: Process of combining data from diverse sources into a unified view.

Data mapping: Technique that defines how data from one data model corresponds to another data model.

Data materialization: Technique that collects data from remote data source and persists a copy the data in a target data storage. This process can also be seen as Extract-Transform-Load (ETL).

Data virtualization: Technique wherein an intermediate component (i.e., data virtualization layer) exposes data available in a remote data sources without creating an copy of the data. The data virtualization layer keeps pointers to the original location of data, so when a data consumer asks for these data, the virtualization layer collects the data from the source and directly serves the data to the consumer.

Ontology: Formal, shared representation of knowledge in a domain.

2.2. Acronyms

CQ: Competency Question

ETL: Extract-Transform-Load

KG: Knowledge Graph

KGC: Knowledge Graph Construction

LOT: Linked Open Terms

OWL: Web Ontology Language

RDF: Resource Description Framework

RDFS: RDF Schema

RML: RDF Mapping Language

SAREF: Smart Applications REFerence

SHACL: Shapes Constraint Language

W3C: World Wide Web Consortium

3. Ontology Development

Ontologies provide the formal representation of the conceptual models that capture the semantics of data, and building on this, the integration of data in the knowledge graph. Ontologies can be developed following different techniques, ranging from manual to fully automated, depending on the characteristics of the data to be integrated in the knowledge graph (e.g., format or schema).

3.1. Standard Development Methodologies

Developing an ontology is a challenging task that requires skills in knowledge management and semantic modelling. To ease this process, a good practice is to follow mature, proven methodologies that provide thorough guidelines and recommend tools that can help in the development of an ontology. An example of these methodologies is Linked Open Terms (LOT) [Poveda-Villalon2022].

LOT is an ontology development methodology that adopts best practices from agile software development. The methodology has been widely used in European projects as well as in the creation of the ETSI SAREF ontology and its extensions. Precisely, with SAREF Ontology ETSI tackled a similar problem in the scope of IoT, where there is a heterogeneous variety of standard data models and protocols. The methodology iterates over a workflow of the following four activities:

1. ontology requirements specification

2. ontology implementation

3. ontology publication, and

4. ontology maintenance.

The workflow starts with the specification of requirements that the ontology must fulfill. To that aim, the methodology requires collecting knowledge from domain experts, but also by analyzing the data sources (e.g., network devices) and schemas for the data (e.g., YANG data models) to be ingested and integrated in the knowledge graph. LOT recommends several approaches such as competency questions (CQs), natural language statements, or tabular information inspired by METHONTOLOGY.

3.2. Automatic Knowledge Extraction from YANG Models

The extraction of knowledge from YANG models could be automated, for example, by analyzing YANG identities to generate controlled vocabularies and taxonomies.

[RFC7950] defines a YANG identity as "globally unique, abstract, and untyped identity", therefore, a relation between a YANG identity and a concept is straightforward. Additionally, YANG identities can inherit from other YANG identities via the "base" statement. These ideas align with the notion of a taxonomy, where concepts are hierarchically linked with other concepts.

To support the creation of knowledge structures like taxonomies or thesauri, the W3C standardized the Simple Knowledge Organization System (SKOS). In such ontology, a concept scheme comprises a set of concepts that can be linked with other concepts via hierarchical and associative relations. Typically, a YANG model containing YANG identities can be represented as an instance of the "skos:ConceptScheme" class. Next, all YANG identities included in a YANG model can be represented as "skos:Concept instances" that are contained in the concept scheme. Lastly, those YANG identities that include the "base" statement, the respective SKOS concept will include a relation "skos:broader" whose range is the SKOS concept representing the parent YANG identity.

4. Knowledge Graph Construction Pipeline

4.1. Knowledge Objects

The intrinsic nature of knowledge graphs is to connect as much knowledge as possible within certain scope---time and/or space. However, not all processes and operations require whole knowledge graphs. For instance, the communication of a piece of telemetry data, organized according to NTF [RFC9232], can be repreented as a subset of the knowledge graph of all measurements.

A knowledge object, as defined in [EERVC], consists in a knowledge graph subset of an arbitrary size---from single atoms to tens or hundreds of triples---that is decorated with metadata to facilitate its contextualization.

Knowledge objects are particularly well suited to enable entities that work with knowledge graphs to communicate to each other knowledge pieces, obtained from their knowledge graphs or newly created from other sources, such as monitoring. It has been demonstrated in [SECDEP].

4.2. Pipeline Steps

The construction of a knowledge graph is supported by a data pipeline that follows the archetypical Extract-Transform-Load (ETL), wherein the raw data is collected from the source(s), transformed, and finally, stored for consumption. The knowledge graph creation pipeline can thus be split into multiple steps as depicted in Figure 1.

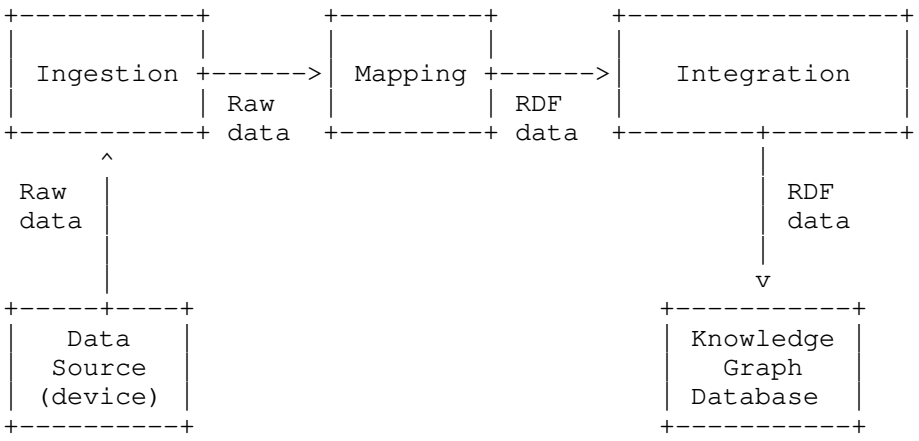


Figure 1: High-level architecture of a Knowledge Graph Construction Pipeline

These steps are the following: ingestion, mapping, and integration.

4.2.1. Ingestion

Represents the first step in the creation of the knowledge graph. This step is realized by means of collectors that ingest raw data from the selected data source. These collectors implement data access protocols which are specific to the technology and type of the data source. For instance, when it comes to network management protocols based on YANG, these protocols can be NETCONF [RFC6241], RESTCONF [RFC8040] and gNMI [GNMI].

Two main types of data sources are identified based on the techniques used to ingest the data, namely, batch and streaming. In the case of batch data sources data are pulled (once or periodically) from the data source.

Regarding streaming data sources, the collector subscribes to a YANG server to receive notifications of YANG data periodically or upon changes in the data source (e.g., a network device whose interface goes down). These subscriptions can be realized, either based on configuration or dynamically, using mechanisms like YANG Push [RFC8641]. But additionally, another common scenario is the use of message broker systems like Apache Kafka for decoupling the ingestion of streams of YANG data [I-D.netana-nmop-yang-message-broker-integration]. Hence, knowledge graph collectors could also support the ingestion of YANG data from these kinds of message brokers.

4.2.2. Mapping

This second step consists at receiving the raw data data from the Ingestion step. Here, the raw data is mapped to the concepts captured in one or more ontologies. By applying these mapping rules, the raw data is semantically annotated and transformed into RDF data. Depending on the nature of the raw data, different techniques can applied.

In the case of (semi-)structured data such as tabular data (e.g., CSV, relational databases) or hierarchical data (e.g., JSON, XML) these mappings can be defined by using declarative languages like RDF Mapping Language (RML) [Iglesias-Molina2023].

RML is a declarative language that is currently being standardized within the W3C Knowledge Graph Construction Community group [W3C-KGC] that allows for defining mappings rules for raw data encoded in semi-structured formats like XML or JSON. The benefits of using a declarative language like RML are twofold: i) the engine that implements the RML rules is generic, thus the mappings rules are decoupled from the code; ii) the explicit representation of mapping and transformation rules as part of the knowledge graph provides data lineage insights that can greatly improve data quality and the troubleshooting of data pipelines. RML is making progress towards becoming a standard, but support of additional YANG encoding formats like CBOR [RFC8949] or Protobuf remains a challenge. The knowledge payload carried by CBOR and/or Protobuf is organized as knowledge objects transmitted by the mapping entities and received by the materialization entities. The use of knowledge objects allows them to easily "cut" knowledge graphs into smaller pieces, transmit them, and "paste" and/or "glue" the pieces onto the destination knowledge graph. Consistency is retained by making the same ontologies be used with the particular knowledge objects.

4.2.3. Integration

This is the final step of the knowledge graph creation. This step receives as an input the knowledge object that contains RDF data generated in the Mapping step, which has easily manageable semantic triples---or quadruples---, as well as metadata to contextualize them and facilicate the incorporation of the knwoledge to the local knowledge graph storage element. At this point, the RDF data can be sent to an RDF triple store like Apache Jena Fuseki [Fuseki] for consumption via SPARQL. But alternatively, this step may transform the RDF data into an LPG structure and store the resulting data in a graph database like Neoj4 [Neo4j]. Similarly, the RDF data could also be transformed into the ETSI NGSI-LD standard [ETSI-GS-CIM-009] and stored in an NGSI-LD Context Broker.

5. Challenges

Ontology development: Time-consuming task that requires skills in knowledge management and conceptual modeling. Additionally, ontology developers should maintain a tight coordination with domain owners and ontology users. Following a standard methodology like LOT provides guidance in the process but still, the development of the ontology requires manual work. Tools that can produce or bootstrap ontologies from existing data models in a semi-automatic, or even automatic, are desirable. In this sense, data models could include explicit semantics in the data models, in the same way that JSON-LD [JSON-LD] or CSVW [CSVW] include metadata indicating which concepts from concepts are referenced by the data.

Pipeline performance: To integrate the raw data from the original data source into the knowledge graph entails several steps as described before. This steps add an extra latency before having the data stored in the knowledge graph for consumption. This latency can be an important limitation for real-time analytics use cases.

Scalability: The knowledge graph must be able to integrate massive amounts of data collected from the network. Distributed and federated architectures can improve the scalability of a global, composable knowledge graph. However, these architectures add complexity to the management of knowledge graph as well as extra latency when federating requests.

Virtualization: The common approach for data integration is by materializing the data in the knowledge graph, which entails duplicating the data. However, this approach presents multiple limitations in terms of data governance and data cadence. Regarding data governance, having copies of the original data hampers keeping track of all the available data. With respect to data cadence, in particular for batch data sources, data are periodically pulled from the source at particular frequency, which might not be optimal depending on the use case. In this sense, data virtualization introduces a new data access technique that can overcome these limitations. With this technique, the knowledge graph defines pointers to the data at the original source, and the KGC pipeline performs the ingestion and mapping of the data, and eventually the delivery of data to the consumer, only when requested on demand.

6. Security Considerations

Access control to data: The knowledge graph becomes an integrator of

data, and, in many cases, sensible. Therefore, data access control mechanisms must be present to ensure that only authorized consumers can discover and access data from the knowledge graph. Access control policies based on roles or attributes are common approaches, but additional aspects like sensitivity of data could be included in the policy.

Integrity and authenticity of mappings: The declaration of mappings of raw data to concepts in ontologies is a critical step in the knowledge graph construction. Unauthorized mappings, or even tampered mappings, can lead to security breaches and anomalies producing a great impact on analytics and machine learning applications that consume data from the knowledge graph. To protect consumers from these scenarios, the knowledge graph must include mechanisms that verify the correctness, authenticity, and integrity of the mappings used in the construction of the graph. Only data owners, as accountable of their data, should be authorized to define and deploy mappings for the knowledge graph construction.

Data provenance: Keeping track of the history of data as they go through the knowledge graph construction pipeline can improve the quality of the data of the knowledge graph. As part of the knowledge graph construction, signatures can be appended to the data [I-D.lopez-opsawg-yang-provenance], can help in verifying that such data come from the golden data source, and therefore, that the data can be trusted.

7. IANA Considerations

This document has no IANA actions.

8. Open Issues

* Should this document provide guidelines for generating URIs of nodes/subjects in the knowledge graph? Take into account there are several levels of abstraction device vs network/service level. For example, the URI that identifies a network interface cannot be generated only from the name of the interface as there could conflicts with other interfaces of other network devices having the same name.

* Implementations? References to examples based on open-source implementations. Integration with YANG-Push-Kafka architecture. Target future hackathons.

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/rfc/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/rfc/rfc8174>.

9.2. Informative References

[ANSA] Pedro Martinez-Julia, Ved P. Kafle, Hitoshi Asaeda., "Application of Category Theory to Network Service Fault Detection. IEEE Open Journal of the Communications Society 5 (2024): 4417-4443.", n.d..

[CSVW] "CSVW - CSV on the Web", n.d., <https://csvw.org>.

[EERVC] Pedro Martinez-Julia, Ved P. Kafle, Hiroaki Harai., "Exploiting External Events for Resource Adaptation in Virtual Computer and Network Systems, IEEE Transactions on Network and Service Management 15 (2018): 555-566.", n.d..

[ETSI-GS-CIM-009] "Context Information Management (CIM); NGSI-LD API", March 2024, <https://www.etsi.org/deliver/etsi_gs/CIM/001_099/009/01.08.01_60/gs_CIM009v010801p.pdf>.

[Fuseki] Apache, "Apache Jena Fuseki", n.d., <https://jena.apache.org/documentation/fuseki2/>.

[GNMI] OpenConfig, "gRPC Network Management Interface (gNMI)", n.d., <https://github.com/openconfig/reference/blob/master/rpc/gnmi/gnmi-specification.md>.

[I-D.ietf-ivy-network-inventory-yang] Yu, C., Belotti, S., Bouquier, J., Peruzzini, F., and P. Bedard, "A Base YANG Data Model for Network Inventory", Work in Progress, Internet-Draft, draft-ietf-ivy-network-inventory-yang-04, 5 November 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-ivy-network-inventory-yang-04>.

[I-D.ietf-netconf-yang-library-augmentedby]
Lin, Z., Claise, B., and I. D. Martinez-Casanueva, "Augmented-by Addition into the IETF-YANG-Library", Work in Progress, Internet-Draft, draft-ietf-netconf-yang-library-augmentedby-01, 21 October 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-yang-library-augmentedby-01>.

[I-D.lopez-opsawg-yang-provenance]
Lopez, D., Pastor, A., Feng, A. H., Birkholz, H., and S. Garcia, "Applying COSE Signatures for YANG Data Provenance", Work in Progress, Internet-Draft, draft-lopez-opsawg-yang-provenance-04, 5 January 2025, <https://datatracker.ietf.org/doc/html/draft-lopez-opsawg-yang-provenance-04>.

[I-D.mackey-nmop-kg-for-netops]
Mackey, M., Claise, B., Graf, T., Keller, H., Voyer, D., and P. Lucente, "Knowledge Graph Framework for Network Operations", Work in Progress, Internet-Draft, draft-mackey-nmop-kg-for-netops-01, 21 October 2024, <https://datatracker.ietf.org/doc/html/draft-mackey-nmop-kg-for-netops-01>.

[I-D.netana-nmop-yang-message-broker-integration]
Graf, T. and A. Elhassany, "An Architecture for YANG-Push to Message Broker Integration", Work in Progress, Internet-Draft, draft-netana-nmop-yang-message-broker-integration-00, 22 April 2024, <https://datatracker.ietf.org/doc/html/draft-netana-nmop-yang-message-broker-integration-00>.

[Iglesias-Molina2023]
Iglesias-Molina, A., "The RML Ontology: A Community-Driven Modular Redesign After a Decade of Experience in Mapping Heterogeneous Data to RDF", The Semantic Web â\200\223 ISWC 2023 , October 2023, <https://doi.org/10.1007/978-3-031-47243-5_9>.

[JSON-LD] W3C, "JSON-LD 1.1: A JSON-based Serialization for Linked Data", July 2020, <https://www.w3.org/TR/json-ld11/>.

[Neo4j] "rdflib-neo4j - RDFLib Store backed by neo4j", n.d., <https://github.com/neo4j-labs/rdflib-neo4j>.

[Poveda-Villalon2022]
Engineering Applications of Artificial Intelligence, "LOT: An industrial oriented ontology engineering framework", May 2022, <https://doi.org/10.1016/j.engappai.2022.104755>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/rfc/rfc6241>.

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <https://www.rfc-editor.org/rfc/rfc7950>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/rfc/rfc8040>.

[RFC8345] Clemm, A., Medved, J., Varga, R., Bahadur, N., Ananthakrishnan, H., and X. Liu, "A YANG Data Model for Network Topologies", RFC 8345, DOI 10.17487/RFC8345, March 2018, <https://www.rfc-editor.org/rfc/rfc8345>.

[RFC8641] Clemm, A. and E. Voit, "Subscription to YANG Notifications for Datastore Updates", RFC 8641, DOI 10.17487/RFC8641, September 2019, <https://www.rfc-editor.org/rfc/rfc8641>.

[RFC8949] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", STD 94, RFC 8949, DOI 10.17487/RFC8949, December 2020, <https://www.rfc-editor.org/rfc/rfc8949>.

[RFC9232] Song, H., Qin, F., Martinez-Julia, P., Ciavaglia, L., and A. Wang, "Network Telemetry Framework", RFC 9232, DOI 10.17487/RFC9232, May 2022, <https://www.rfc-editor.org/rfc/rfc9232>.

[SECDEP] Ana Hermosilla, Jose Manuel ManjÃ³n-CÃ¡liz, Pedro Martinez-Julia, Antonio Pastor, Jordi Ortiz, Diego R. Lopez, Antonio Skarmeta., "Secure deployment of third-party applications over 5G-NFV ML-empowered infrastructures, the 7th International Conference on Mobile Internet Security (MobiSec ’23), Dec 19-21, 2023, Okinawa, Japan.", n.d..

[TKDP] Pedro Martinez-Julia, Ved P. Kafle, Hitoshi Asaeda., "Telemetry Knowledge Distributed Processing for Network Digital Twins and Network Resilience. NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium (2023): 1-6.", n.d..

[W3C-KGC] W3C, "Knowledge Graph Construction Community Group", n.d., <https://www.w3.org/community/kg-construct/>.

Appendix A. NETCONF Data Sources

This appendix presents a scenario that demonstrates the construction of a knowledge graph based on YANG data collected from a NETCONF server. In particular, the scenario tackles the creation of a data catalog based on a knowledge graph that keeps registry of the YANG data sources and the YANG data models that they implement.

As described in [I-D.ietf-netconf-yang-library-augmentedby], data catalog implementations backed by knowledge graphs provide powerful solutions that can easily incorporate additional context to the catalog. As an evolution of the YANG Catalog service, the resulting knowledge graph facilitates the navigation across dependencies of YANG modules, but more importantly, enables the combination of these data with other data silos such as the network topology [RFC8345] or network hardware inventory [I-D.ietf-ivy-network-inventory-yang].

To create a knowledge graph that supports the data catalog, the proposed approach is based on collecting data from the YANG Library from devices running in the network, in this case, from a NETCONF server. For this, the RML engine queries the NETCONF server to retrieve the YANG Library data, and then, applies the RML mappings to transform the YANG data into RDF according to the target ontology.

This prototype was conducted as part of the paper "Declarative Construction of Knowledge Graphs from NETCONF Data Sources" sent to the Semantic Web Journal (currently under review): https://www.semantic-web-journal.net/content/declarative-construction-knowledge-graphs-netconf-data-sources-0

A.1. Prototype Architecture

A high-level architecture of the prototype that validates the implementation is shown below:

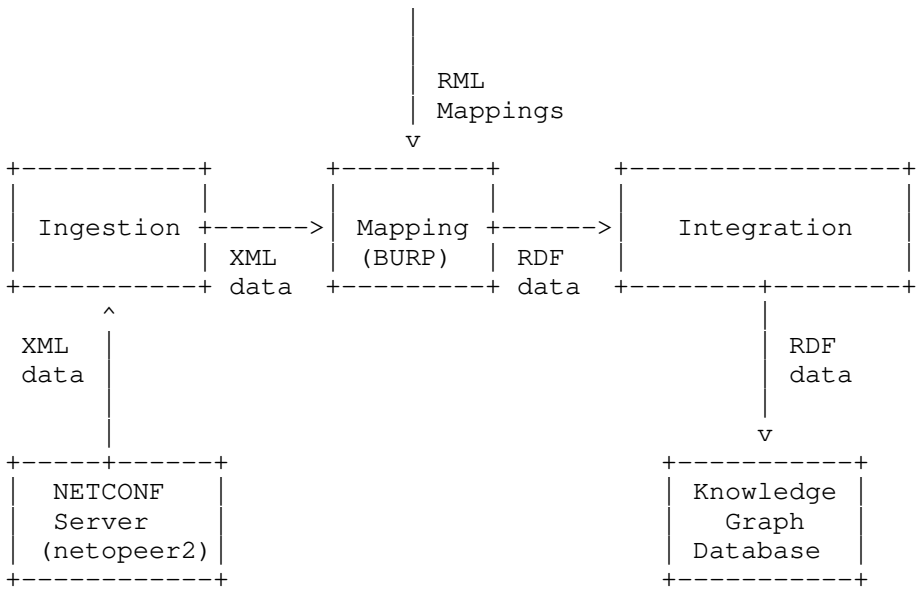


Figure 2: Architecture of prototype to construct knowledge graph from NETCONF data source

BURP was selected as the open-source implementation of an RML engine that was chosen for this prototype. The NETCONF server is emulated using the netopeer2.

A.2. Target Ontology

The YANG Library Ontology was developed to represent the implementation details of YANG module and submodules, along with their interdependencvies, in the different datastores of YANG server. The ontology was developed following the LOT methodology and is publicly available at: https://w3id.org/yang/library

The code of the ontology and all related artifacts are publicly available on GitHub: https://github.com/candil-data-fabric/yang-library-ontology

A.3. KGC Pipeline

In addition to the YANG Library Ontology, the YANG Server Ontology was developed to represent YANG data sources such as NETCONF servers and operations to retrieve data from them such as queries or subscriptions. Similarly, this ontology was developed following the LOT methodology and is publicly available at: https://w3id.org/yang/server

The code of the ontology and all related artifacts are publicly available on GitHub: https://github.com/candil-data-fabric/yang-server-ontology

The YANG Server Ontology is used in combination with the RML vocabulary to describe the access to YANG servers, from which the collected data are transformed into RDF. In this sense, BURP was extendended to support the ingestion of YANG data from NETCONF servers using NETCONF queries.

The following subsections include excerpts of the raw XML data (Figure 3), RML mappings (Figure 4), and final RDF data (Figure 3). The complete examples can be found on: https://github.com/candil-data-fabric/yang-library-ontology/tree/main/knowledge-graph/xpath

A.3.1. Raw data

```
<modules-state xmlns="urn:ietf:params:xml:ns:yang:ietf-yang-library"
  xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <module-set-id>1</module-set-id>
  <module>
    <name>ietf-yang-patch</name>
    <revision>2017-02-22</revision>
    <schema>file:///etc/sysrepo/yang/ietf-yang-patch@2017-02-22.yang</schema>
    <namespace>urn:ietf:params:xml:ns:yang:ietf-yang-patch</namespace>
    <conformance-type>import</conformance-type>
  </module>
  <module>
    <name>ietf-ip</name>
    <revision>2018-02-22</revision>
    <schema>file:///etc/sysrepo/yang/ietf-ip@2018-02-22.yang</schema>
    <namespace>urn:ietf:params:xml:ns:yang:ietf-ip</namespace>
    <conformance-type>implement</conformance-type>
  </module>
</modules-state>
```

Figure 3: Excerpt of YANG Library data collected from a NETCONF server

A.3.2. Mappings

```
@prefix yl: <https://w3id.org/yang/library#> .
@prefix ys: <https://w3id.org/yang/server#> .
@prefix rml: <http://w3id.org/rml/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix core: <https://ontology.unifiedcyberontology.org/uco/core/> .
@prefix dcterms: <http://purl.org/dc/terms/> .
@prefix observable: <https://ontology.unifiedcyberontology.org/uco/observable/> .
```

```
@base <https://netconf-rml-demo.org/> .

# Connection details to NETCONF server
<netconf-server-1> a ys:NetconfServer ;
    ys:socketAddress <netconf-server-1/socket-address> ;
    ys:serverAccount <netconf-server-1/account> ;
    ys:hostKeyVerification "false" ;
    ys:capability ys:XpathCapability ,
                  ys:YangLibrary1.0 .

<netconf-server-1/datastores/operational> a ys:OperationalDatastore ;
    ys:server <netconf-server-1> .

<netconf-server-1/datastores/running> a ys:RunningDatastore ;
    ys:server <netconf-server-1> .

<netconf-server-1/socket-address> a observable:SocketAddress ;
    observable:addressValue "localhost:830" .

<netconf-server-1/account> a ys:ServerAccount ;
    ys:username "netconf" ;
    core:hasFacet <netconf-server-1/account/authentication> .

<netconf-server-1/account/authentication> a observable:AccountAuthenticationFacet
 ;
    observable:password "netconf" ;
    observable:passwordType "plain-text" .

<filters/xpath/yang-library> a ys:XPathFilter ;
    ys:xpathValue "/yanglib:modules-state";
    ys:namespace [ a ys:Namespace ;
      ys:namespacePrefix "yanglib" ;
      ys:namespaceURL "urn:ietf:params:xml:ns:yang:ietf-yang-library" ;
    ];
.

<#TriplesMap> a rml:TriplesMap;
  rml:logicalSource [ a rml:LogicalSource;
    rml:source [ a ys:Query, rml:Source ;
      ys:sourceDatastore <netconf-server-1/datastores/operational> ;
      ys:filter <filters/xpath/yang-library>
    ];
    rml:referenceFormulation [ a ys:NetconfQuerySource ;
      rml:namespace [ a rml:Namespace ;
        rml:namespacePrefix "yanglib" ;
        rml:namespaceURL "urn:ietf:params:xml:ns:yang:ietf-yang-library" ;
      ];
    ];
    rml:iterator "/yanglib:modules-state/yanglib:module";
```

```
  ];
  rml:subjectMap [ a rml:SubjectMap;
    rml:template "http://example.org/module/{yanglib:name/text()}:{yanglib:revisi
on/text()}";
    rml:class yl:Module;
  ];
  rml:predicateObjectMap [ a rml:PredicateObjectMap;
    rml:predicateMap [ a rml:PredicateMap;
      rml:constant yl:moduleName;
    ];
    rml:objectMap [ a rml:ObjectMap;
      rml:reference "yanglib:name/text()";
      rml:datatype xsd:string;
    ];
  ];
  rml:predicateObjectMap [ a rml:PredicateObjectMap;
    rml:predicateMap [ a rml:PredicateMap;
      rml:constant yl:revisionDate;
    ];
    rml:objectMap [ a rml:ObjectMap;
      rml:reference "yanglib:revision/text()";
      rml:datatype xsd:date;
    ];
  ];
  rml:predicateObjectMap [ a rml:PredicateObjectMap;
    rml:predicateMap [ a rml:PredicateMap;
      rml:constant yl:namespace;
    ];
    rml:objectMap [ a rml:ObjectMap;
      rml:reference "yanglib:namespace/text()";
      rml:datatype xsd:anyURI;
    ];
  ];
.
```

Figure 4: RML mappings that collect YANG Library from a NETCONF server and map them to the YANG Library Ontology

A.3.3. RDF data

```
<http://example.org/module/ietf-ip:2018-02-22>
  <https://w3id.org/yang/library#moduleName>
  "ietf-ip"^^<http://www.w3.org/2001/XMLSchema#string> .
<http://example.org/module/ietf-ip:2018-02-22>
  <https://w3id.org/yang/library#revisionDate>
  "2018-02-22"^^<http://www.w3.org/2001/XMLSchema#date> .
<http://example.org/module/ietf-ip:2018-02-22>
  <https://w3id.org/yang/library#namespace>
  "urn:ietf:params:xml:ns:yang:ietf-ip"^^<http://www.w3.org/2001/XMLSchema#anyURI
> .
<http://example.org/module/ietf-ip:2018-02-22>
  <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
  <https://w3id.org/yang/library#Module> .
<http://example.org/module/ietf-yang-patch:2017-02-22>
  <https://w3id.org/yang/library#moduleName>
  "ietf-yang-patch"^^<http://www.w3.org/2001/XMLSchema#string> .
<http://example.org/module/ietf-yang-patch:2017-02-22>
  <https://w3id.org/yang/library#revisionDate>
  "2017-02-22"^^<http://www.w3.org/2001/XMLSchema#date> .
<http://example.org/module/ietf-yang-patch:2017-02-22>
  <https://w3id.org/yang/library#namespace>
  "urn:ietf:params:xml:ns:yang:ietf-yang-patch"^^<http://www.w3.org/2001/XMLSchem
a#anyURI> .
<http://example.org/module/ietf-yang-patch:2017-02-22>
  <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
  <https://w3id.org/yang/library#Module> .
```

Figure 5: Excerpt of RDF triples generated using the RML mappings and the YANG Library data

Acknowledgments

This document is based on work partially funded by the EU Horizon Europe projects aerOS (grant agreement no. 101069732) and ROBUST-6G (grant agreement no. 101139068).

The authors would like to thank Med, Benoit, Lionel, and Thomas for their review and valuable comments.

Authors' Addresses

Ignacio Dominguez Martinez-Casanueva
Telefonica
Email: ignacio.dominguezmartinez@telefonica.com

Lucia Cabanillas
Telefonica
Email: lucia.cabanillasrodriguez@telefonica.com

Pedro Martinez-Julia
NICT
Email: pedro@nict.go.jp

Network Management Operations
Internet-Draft
Intended status: Informational
Expires: 16 November 2025

L. Tailhardat
Orange Research
R. Troncy
EURECOM
Y. Chabot
F. Ramparany
P. Folz
Orange Research

15 May 2025

Knowledge Graphs for Enhanced Cross-Operator Incident Management and Network Design
draft-tailhardat-nmop-incident-management-noria-02

Abstract

Operational efficiency in incident management on telecom and computer networks requires correlating and interpreting large volumes of heterogeneous technical information. Knowledge graphs can provide a unified view of complex systems through shared vocabularies. YANG data models enable describing network configurations and automating their deployment. However, both approaches face challenges in vocabulary alignment and adoption, hindering knowledge capitalization and sharing on network designs and best practices. To address this, the concept of a IT Service Management (ITSM) Knowledge Graph (KG) is introduced to leverage existing network infrastructure descriptions in YANG format and enable abstract reasoning on network behaviors. The key principle to achieve the construction of such ITSM-KG is to transform YANG representations of network infrastructures into an equivalent knowledge graph representation, and then embed it into a more extensive data model for Anomaly Detection (AD) and Risk Management applications. In addition to use case analysis and design pattern analysis, an experiment is proposed to assess the potential of the ITSM-KG in improving network quality and designs.

About This Document

This note is to be removed before publishing as an RFC.

The latest revision of this draft can be found at https://genears.github.io/draft-tailhardat-nmop-incident-management-noria/draft-tailhardat-nmop-incident-management-noria.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-tailhardat-nmop-incident-management-noria/.

Discussion of this document takes place on the Network Management Operations Working Group mailing list (mailto:nmop@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/nmop/. Subscribe at https://www.ietf.org/mailman/listinfo/nmop/.

Source for this draft and an issue tracker can be found at https://github.com/genears/draft-tailhardat-nmop-incident-management-noria.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 16 November 2025.

Copyright Notice

Table of Contents

1. Introduction

Incident management on telecom and computer networks, whether it is related to infrastructure or cybersecurity issues, requires the ability to simultaneously and quickly correlate and interpret a large number of heterogeneous technical information sources. Knowledge graphs, by structuring heterogeneous data through shared vocabularies, enable providing a unified view of complex technical systems, their ecosystem, and the activities and operations related to them (see [I-D.marcas-nmop-knowledge-graph-yang] and [NORIA-O-2024]). Using such formal knowledge representation allows for a simplified interpretation of networks and their behavior, both for NetOps & SecOps teams and artificial intelligence (AI) algorithms (e.g. anomaly detection, root cause analysis, diagnostic aid, situation summarization), and paves the way, in line with the Network Digital Twin vision [I-D.irtf-nmrg-network-digital-twin-arch], for the development of tools for detecting and analyzing complex network incident situations through explainable, actionable, and shareable models (see [FOLIO-2018], [SLKG-2023], and [GPL-2024]).

However, despite potential benefits of using knowledge graphs, these are not mainstream yet in commercial network deployment systems and decision support systems (see [NORIA-UI-2024] for more on the decision support systems perspective). YANG is a widely used standard among operators for describing network configurations and automating their deployment. Using YANG representations in the form of a KG, as suggested in [I-D.marcas-nmop-knowledge-graph-yang], would minimize the effort required to adapt network management tools towards the unified vision and applications evoked above. The lack of alignment between various YANG models on key concepts (e.g. for describing network topology) is, however, hindering this evolution [I-D.boucadair-nmop-rfc3535-20years-later].

Furthermore, although [I-D.netana-nmop-network-anomaly-lifecycle] addresses the capitalization of incident management knowledge through a YANG model, it can be observed that the overall scope of YANG models does not naturally cover the description of the networks' ecosystem (e.g. physical equipment location, operator organization, supervision systems) or the description of network operations from an IT service management (ITSM) perspective (e.g. business processes and design rules used by the company, scheduled modification operations, remediation actions performed during incident handling). As a consequence, the continuous improvement of network quality & designs requires additional data cross-referencing operations to properly contextualize incidents and learn from remediation actions taken (e.g. analyzing intervention technicians' verbatim, comparing actions performed on similar incidents but occurring on different networks). As a result of these additional efforts of contextualization, the capitalization of knowledge typically remains confined at the level of each network operator. This, in turn, hinders the sharing of information within the community of researchers and system designers regarding failure modes and best practices to adopt, considering the concept of overall improvement of IT systems and the Internet.

Realizing an ITSM knowledge graph for network deployment, anomaly detection and risk management applications has been studied for several years in the Semantic Web community (i.e. knowledge representation and automated reasoning leveraging Web technologies such as [RDF], [RDFS], [OWL], and [SKOS]). Among other examples: the DevOpsInfra ontology [DevOpsInfra-2021] allows for describing sets of computing resources and how they are allocated for hosting services; the NORIA-O ontology [NORIA-O-2024] allows for describing a network infrastructure & ecosystem, its events, diagnosis and repair actions performed during incident management. Assuming the continuous integration into a knowledge graph of data from ticketing systems, network monitoring solutions, and network configuration management databases, we remark that the resulting knowledge graph (Figure 1) implicitely holds the necessary information to (automatically) learn

incident contexts (i.e. the network topology, its set of states and set of events prior to the incident) and remediation procedures (i.e. the set of actions and network configuration changes carried-out to resolve the incident).

â\224\214â\224\200â\224\200â\224\200Incident contextâ\224\200â\224\220
â\224\202 â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\220
â\224\202
â\224\202 â\224\202skos:Conceptâ\224\202 â\224\202
â\224\202 â\224\224â\224\200â\224¬â\224¬â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\230
â\224\202
â\224\202 <server> â\224\202
â\224\202 â\226² â\224\202
â\224\202 â\224\202 â\224\202
â\224\202 resourceType â\224\202
â\224\202 â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\220 â\224\202 â\224\202
â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\220
â\224\202 â\224\202Resourceâ\224\202 â\224\202
â\224\202 â\224\202TroubleTicketâ\224\202
â\224\202 â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224¬â\224¬â\224\230 â\224\202 â\224\202 â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â\224¬â\224¬â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\230
â\224\202 â\224\202â\224\202 â\224\202
â\224\202 â\224\202â\224\202
â\224\202 <ne_2>â\224\200â\224\200<ne_1>â\227\204â\224\200â\224\200troubleTicketRelatedResourceâ\224\200â\224\200<incident_01>
â\224\202 â\224\202 â\224\202 â\224\202
â\224\202 â\224\202 â\224\202 â\224\202 problemCategory
â\224\202<ne_5>â\224\200â\224\200<ne_4>â\224\200â\224\200â\224\200â\224\200â\224¼â\224\200â\224\200<ne_3>â\224\200â\224\200â\224\200â\224\200<log_2>
â\224\202 â\224\202
â\224\202 â\224\202 â\224\202 â\224\202 â\224\202
â\224\202 â\226¼
â\224\202 â\224\202 â\224\202 â\224\202
â\224\202 <packet-loss>
â\224\202 <log_3> â\224\202 <ne_6> â\224\202
â\224\202â\224\202
â\224\202 â\224\202 â\224\202
â\224\214â\224\200â\224\200â\224\200â\224\200â\224´â\224´â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\220
â\224\202 logOriginatingManagedObject â\224\202 â\224\202skos:Conceptâ\224\202
â\224\202 â\224\202 â\224\202
â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\230
â\224\202 â\226¼ â\224\202
â\224\202 <log_1>â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\220 â\224\202
â\224\202 â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224´â\224´â\224\220 dcterms:type â\224\202

Figure 1: Learning an incident signature seen as a classification model that is trained on the relationship of the incident context (i.e. a subgraph centered around a Resource entity concerned by a given TroubleTicket) to the problem class defined at the TroubleTicket entity level. Arrows are for object properties (owl:ObjectProperty), double line edges are for object class relationships (rdf:type).

By going a step further, we notice that a generic understanding of incident context can be extracted and shared among operators from knowledge graphs. Indeed, a knowledge graph, being an instantiation of shared vocabularies (e.g. RDFS/OWL ontologies and controlled vocabularies in SKOS syntax), sharing incident signatures can be done without revealing infrastructure details (e.g. hostname, IP address), but rather the abstract representation of the network (i.e. the class of the knowledge graph entities and relationships, such as "server" or "router", and or "IPoWDM link").

The remainder of this document is organized as follows. Firstly, the concept of an ITSM-KG is introduced in Section 3 towards leveraging existing network infrastructure descriptions in YANG format and enabling abstract reasoning on network behaviors. The relation of the ITSM-KG proposal to the Digital Map [I-D.havel-nmop-digital-map-concept] is notably discussed in this section. Secondly, strategies for the ITSM-KG construction are discussed in Section 4. This include YANG models transformation in Section 4.1, implementing alignments of models with the ITSM-KG in Section 4.2, and knowledge graph construction pipeline designs in Section 4.3. The Section 4.3 notably focuses on addressing the handling of event data streams and providing a unified view for different stakeholders, also known as the data federation architecture. Finally, an experiment is proposed in Section 5 to assess the potential of the ITSM-KG in improving network quality and designs. The implementation status related to this document is also reported in this section.

2. Conventions and Definitions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. An ITSM-KG for Learning and Sharing Network Behavioral Models

3.1. Principles

As evoked in Section 1, a detailed characterization of network behavior requires combining several facets of data related both to the configuration of the networks and to their lifecycle, as well as the ecosystem in which they are operated. In this document, we will consider the following fundamental definitions as a means to achieve the combination of all these facets of data in a convenient way, regardless of their origin, for operational efficiency in incident management and change management with the aid of AI tools:

ITSM-KG: A knowledge graph in RDFS/OWL syntax tha enables change management activities, anomaly detection, and risk analysis at the organizational level by combining heterogeneous data sources from the configuration data of the network's structural elements, events occurring on this network, and any other data useful to the business for the effective management of the services provided by this network.

ONTO-ITSM: For a given ITSM-KG, the RDFS/OWL ontology that structures the ITSM-KG.

ONTO-YANG-MODEL: For a given YANG model, its equivalent RDFS/OWL representation.

ONTO-META: An ontology that contributes to structuring some ITSM-KG, regardless of the specifics of a given application domain or ITSM-KG instance, in the sense that it provides an abstract IT Service Management model (i.e. it holds generic concept and property definitions for realizing IT Service Management activities).

ONTO-LINKER: For a given (set of) ONTO-YANG-MODEL and a given ONTO-META, the implementation of the equivalence relationships between the key concepts and key properties of the (set of) ONTO-YANG-MODEL and ONTO-META.

Based on these definitions, which will be discussed in more detail later in this document, Figure 1 can be seen as an illustration of ITSM-KG from which a subgraph has been extracted, allowing for incident situation to be analyzed through querying. For example, close to ideas from [I-D.netana-nmop-network-anomaly-lifecycle], querying the evolution of network entities states from the ITSM-KG during some incident remediation stage could bring to identify the causal graph underlying incident resolution. As the querying would go through the ONTO-ITSM, the causal graph would de-facto be an abstraction of the situation, thereby enabling knowledge capitalization and sharing for similar incidents that could occur later.

3.2. Relation to the Digital Map

Similar to the concept of ITSM-KG discussed in this document, the concept of Digital Map discussed in [I-D.havel-nmop-digital-map-concept] emphasizes the need to structure heterogeneous data describing networks in order to simplify network management operations through unified access to this data. The ITSM-KG can be seen as a meta-knowledge graph that extends the Digital Map concept by adding information about the lifecycle of infrastructures and services, as well as the context of their usage. These

additional pieces of information are considered essential for learning shareable activity models of systems.

To clarify this positioning, the following lists (Section 3.2.1, Section 3.2.2, and Section 3.2.3) reflect the compliance of the meta-KG concept with the Digital Map Requirements defined in [I-D.havel-nmop-digital-map-concept]. A symbol to the right of each requirement name indicates the nature of compliance: *+* for compatibility, */* for partial satisfaction, *-* for non-compliance with the requirement. A comment is provided as necessary.

3.2.1. Core Requirements

+ REQ-BASIC-MODEL-SUPPORT: nothing to report (n.t.r.)

+ REQ-LAYERED-MODEL: n.t.r.

/ REQ-PROG-OPEN-MODEL: Partially satifying the requirement as the concept of meta-KG mainly relate to the knowledge representation topic rather than to the platform running the Digital Map service on top of the meta-knowledge graph.

/ REQ-STD-API-BASED: Same remark as for REQ-PROG-OPEN-MODEL.

+ REQ-COMMON-APP: n.t.r.

+ REQ-SEMANTIC: n.t.r.

+ REQ-LAYER-NAVIGATE: n.t.r.

+ REQ-EXTENSIBLE: Knowledge graphs implicitly satisfy this requirement, notably with OWL [OWL] and SKOS [SKOS] constructs if considering RDF knowledge graphs for the meta-KG (e.g. owl:sameAs to relate a meta-KG entity to some other entity of another knowledge graph, owl:equivalentClass to link concepts and properties used to interpret the meta-KG to concepts and properties from other data models, skos:inScheme to group new items of a controled-vocabulary as part of a skos:ConceptScheme).

+ REQ-PLUGG: Same remark as for REQ-EXTENSIBLE.

+ REQ-GRAPH-TRAVERSAL: This capability is naturally enabled as the meta-KG concept involves using a graph data structure.

3.2.2. Design Requirements

- REQ-TOPO-ONLY: Requirement not satisfied as the meta-KG involves

to have more than topological data to interpret and contextualize the network behavior.

- REQ-PROPERTIES: Same remark as for REQ-TOPO-ONLY.

- REQ-RELATIONSHIPS: Same remark as for REQ-TOPO-ONLY.

+ REQ-CONDITIONAL: Native, notably considering the expressiveness of SPARQL [SPARQL11-QL] if using the Semantic Web protocol stack to run the meta-KG concept.

+ REQ-TEMPO-HISTO: n.t.r.

3.2.3. Architectural Requirements

+ REQ-DM-SCALES: This capability applies as we can use data aggregation at the graph level (Figure 10 and Figure 11 compared to Figure 8 and Figure 9), aggregation without loss of information (Figure 10 and Figure 11), and load balancing (horizontal scaling) by partitioning the meta-KG (Figure 12). Further, ease of integration is enabled thanks to existing standard graph data access protocols (e.g. SPARQL Federated Queries [SPARQL11-FQ], as illustrated in Figure 12).

/ REQ-DM-DISCOVERY: Same remark as for REQ-PROG-OPEN-MODEL.

4. Strategies for the ITSM-KG Construction

In this section, we firstly define in Section 4.1 two YANG-based data transformation scenario, namely the YANG-KG-SEMANTIC-EQUIVALENCE and YANG-KG-SEMANTIC-GENERALIZATION scenarios. The YANG-KG-SEMANTIC-GENERALIZATION scenario is then used as a basis in Section 4.2 to illustrate strategies to reuse YANG models transformed in RDFS/OWL syntax in a higher-level ontology that would structure the ITSM-KG. Finally, two Extract-Transform-Load (ETL) pipeline approaches and a data federation architecture are presented in Section 4.3 to meet the needs of constructing and exploiting the ITSM-KG.

4.1. From YANG-based Configurations to Meta-Knowledge Graph

In the following, we consider the use of Semantic Web technologies as the foundation for representing data in the form of a knowledge graph. We also assume the ability to transform a description of configurations and network infrastructures expressed accordingly to a given (set of) YANG model(s) into a knowledge graph representation.

For the realization of this data transformation, we identify the following scenarios:

YANG-KG-SEMANTIC-EQUIVALENCE: The ontology structuring the target knowledge graph is an exact equivalence of the many YANG models organizing the configuration data.

YANG-KG-SEMANTIC-GENERALIZATION: The ontology structuring the target KG is a generalization of the YANG models organizing the configuration data.

We note that the YANG-KG-SEMANTIC-EQUIVALENCE case requires a significant knowledge engineering effort to align all YANG models into a coherent ontology with a sufficient level of abstraction to enable the discovery and analysis of emergent behavioral models of networks independently of local configuration specifics. However, this case has the advantage of being relatively easy to implement based on the available configuration data of an operator, for example, by implementing [RML] rules for constructing a knowledge graph from this data.

For the YANG-KG-SEMANTIC-GENERALIZATION case, we observe that the transformation effort involves:

1. Being able to transform YANG models into their RDFS/OWL equivalent to provide a consistent interpretation of configuration data in a knowledge graph that aligns with each data source.

2. Being able to provide a generalized interpretation of these transformed YANG models by identifying alignments between key concepts in these models and those in a more expressive ontology.

As an example, the YANG-KG-SEMANTIC-GENERALIZATION case could involve wanting to integrate Service and Network topology data, matching the Network Topologies [RFC8345] and Service Assurance [RFC9418] YANG data models, into a knowledge graph structured by the NORIA-O ontology [NORIA-O-2024].

Although identifying alignments in the YANG-KG-SEMANTIC-GENERALIZATION case may appear non-trivial for "constructor" YANG models, it is worth noting that the design of YANG models generally relies on principles of concept hierarchies and reuse of common concepts between models to promote model interoperability, as is the case with the Abstract Network Model of [RFC8345]. Therefore, the task of identifying alignments can theoretically benefit from these design principles.

In continuity of the above RFC8345 / NORIA-O example, providing an alignment may mean asserting a semantic equivalence between the RDFS/OWL representation of the "node" concept from [RFC8345] with the "noria:Resource" concept from [NORIA-O-2024]. Examples of approaches for linking ontologies are provided in Section 4.2.

4.2. Implementing Alignments of Model-Specificities to a Multi-Faceted Knowledge Graph

Building on the previously defined YANG-KG-SEMANTIC-GENERALIZATION scenario, this section presents two approaches to construct the structuring ontology of the ITSM-KG by combining YANG models translated into RDFS/OWL and a meta-ontology enabling the analysis of the operational context of the network lifecycle. As techniques for identifying alignments between data models is beyond the scope of this document, we refer interested readers to specialized literature in this field, such as [ONTO-MATCH-2022].

To present the approaches, we assume the ability to convert a given YANG model into its ONTO-YANG-MODEL (i.e. its equivalent RDFS/OWL representation). The code snippet in Figure 2 is a fictional example of translating the "node" concept from [RFC8345] into its RDFS/OWL equivalent.

```
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

<urn:ietf:params:xml:ns:yang:ietf-network#node>
  rdf:type owl:Class ;
  rdfs:comment  "The inventory of nodes of this network." ;
.
```

Figure 2: Snippet of the ONTO-YANG-MODEL describing the 'node' concept from RFC8345 into its RDFS/OWL equivalent, in Turtle syntax.

The following sub-sections build on the ONTO-YANG-MODEL example from Figure 2.

4.2.1. The Network of Ontologies Approach

The network of ontologies approach is a common practice in the field of knowledge engineering and Semantic Web technologies. The principle involves assembling vocabularies from different domains to form a coherent set, for example to infer - through graph traversal or reasoning - relationships between entities in the graph, starting from a concept defined in one of the vocabularies and leading to an

instance of a concept from another vocabulary.

In our example, the code snippet of Figure 3 implements the ONTO-ITSM by importing concepts from the ONTO-YANG-MODEL (Figure 2) and concepts from the ONTO-META (Figure 4). An additional import in Figure 5 relates to the ONTO-LINKER.

```
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

<https://example.com/ontologies/itsm/>
  rdf:type owl:Ontology ;
  owl:imports
    # ===> Import of one of the ONTO-YANG-MODEL <===
    <https://example.com/ontologies/ietf-network-topology> ,
    # ===> Import of the ONTO-META <===
    <https://w3id.org/noria/ontology/> ,
    # ===> Import of the ONTO-LINKER definitions <===
    <https://example.com/ontologies/ietf-noria-linker> ;
.
```

Figure 3: The implementation of the ONTO-ITSM to structure the relation of ONTO-YANG-MODEL(s) with ONTO-META, in Turtle syntax.

```
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

@prefix seas: <https://w3id.org/seas/>.  # Smart Energy Aware Systems
@prefix bot:  <https://w3id.org/bot#> .  # Building Topology Ontology
@prefix observable:  # Unified Cybersecurity Ontology (UCO)
  <https://unifiedcyberontology.org/ontology/uco/observable#> .
@prefix log: <https://w3id.org/sepses/ns/log#> .  # a.k.a. SLOGERT

@prefix noria: <https://w3id.org/noria/ontology/> .

noria:Resource
    rdf:type owl:Class ;
    rdfs:label "Resource" ;
    rdfs:comment """General resource record of the Communication Device
      kind from the logistics park. It is a managed entity that can be
      either Physical or Virtual."""@en ;
    rdfs:subClassOf noria:StructuralElement ;
    rdfs:subClassOf
        seas:System,
        seas:CommunicationDevice,
        bot:Element ,
        observable:Device ,
        log:Host ;
    rdfs:isDefinedBy noria: ;
.
```

Figure 4: Snippet of the ONTO-META describing the 'noria:Resource' concept from NORIA-O v0.3, in Turtle syntax.

```
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix noria: <https://w3id.org/noria/ontology/> .

noria:Resource
  owl:equivalentClass <urn:ietf:params:xml:ns:yang:ietf-network#node> ;
.
```

Figure 5: Snippet of the ONTO-LINKER to relate ONTO-YANG-MODEL definition(s) with ONTO-META definition(s), in Turtle syntax.

As a result, querying any ITSM-KG structured by the ONTO-ITSM, as shown in Figure 6, enables retrieving entities of the ITSM-KG using ONTO-META concepts, even if entities are described with ONTO-YANG-MODEL concepts.

```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX noria: <https://w3id.org/noria/ontology/>

SELECT ?res

WHERE {
  # Pattern for the base class from ONTO-META
  # or any equivalent class from ONTO-YANG-MODEL
  ?resClass (owl:equivalentClass|^owl:equivalentClass)* noria:Resource .

  # Pattern to retrieve instances from the ITSM-KG
  ?res rdf:type ?resClass .
}
```

Figure 6: Snippet to retrieve entities of the ITSM-KG assuming the relatedness of ONTO-META concepts with ONTO-YANG-MODEL concepts, in SPARQL syntax.

4.2.2. Explicit Linking in the ONTO-META

In this approach, we assume that we have the means to evolve ONTO-META, which allows for the implementation of equivalence relationships between the concepts of ONTO-META and ONTO-YANG-MODEL directly within ONTO-META, as shown in Figure 7.

In this sense, ONTO-ITSM is part of ONTO-META, and ONTO-LINKER is within ONTO-META. The query in Figure 6 applies here as well and will yield the same results.

```
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

@prefix seas: <https://w3id.org/seas/>.  # Smart Energy Aware Systems
@prefix bot:  <https://w3id.org/bot#> .  # Building Topology Ontology
@prefix observable:  # Unified Cybersecurity Ontology (UCO)
  <https://unifiedcyberontology.org/ontology/uco/observable#> .
@prefix log: <https://w3id.org/sepses/ns/log#> .  # a.k.a. SLOGERT

@prefix noria: <https://w3id.org/noria/ontology/> .

<https://w3id.org/noria/ontology/>
  a owl:Ontology ;
  # ===> Import of one of the ONTO-YANG-MODEL <===
  <https://example.com/ontologies/ietf-network-topology> .

noria:Resource
    rdf:type owl:Class ;
    rdfs:label "Resource" ;
    rdfs:comment """General resource record of the Communication Device
      kind from the logistics park. It is a managed entity that can be
      either Physical or Virtual."""@en ;
    rdfs:subClassOf noria:StructuralElement ;
    rdfs:subClassOf
        seas:System,
        seas:CommunicationDevice,
        bot:Element ,
        observable:Device ,
        log:Host ;
    rdfs:isDefinedBy noria: ;
    # ===> Explicit linking to ONTO-YANG-MODEL <===
    owl:equivalentClass <urn:ietf:params:xml:ns:yang:ietf-network#node>
.
```

Figure 7: Snippet of the ONTO-META describing the 'noria:Resource' concept from NORIA-O v0.3 with added linking to ONTO-YANG-MODEL, in Turtle syntax.

4.3. Extract-Transform-Load Pipelines for the ITSM-KG

Based on [I-D.marcas-nmop-knowledge-graph-yang] and [NORIA-DI-2023], which present the technical means to implement a pipeline for constructing the ITSM-KG, this section focuses on two complementary viewpoints: Section 4.3.1 the management of streaming data such as alarms and logs, and Section 4.3.2 the deployment of a federated data architecture when various technical foundations or business units are involved in providing the ITSM-KG.

From the perspective of the Digital Map Requirements (Section 3.2), the Figure 10, Figure 11 and Figure 12 particularly address the REQ-DM-SCALES requirement.

4.3.1. Handling Event Streams

The following figures illustrate different scenarios for constructing a ITSM-KG through an Extract-Transform-Load (ETL) data integration pipeline.

Figure 8 illustrates a common design pattern providing the capability to record event streams into a knowledge graph, such as an ITMS-KG if considering that event data are mapped to ONTO-META concepts and network entities to ONTO-YANG-MODEL concepts. The Figure 9 provides an example of the resulting representation in the form of a knowledge graph.


```
            â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224
\220  â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\220  â\224\214â\224\200â\224\200â\224\200â\224\200â\224
\200â\224\200â\224\220  â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\200â\224\220  â\224\214â\224\200â\224\200â\224\200â\224
\200â\224\200â\224\200â\224\220
   â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\220  â
\224\202      â\224\202  â\224\202 Stream  â\224\202  â\224\202      â\224\202  â
\224\202 Stream â\224\202  â\224\202â\224\214â\224\200â\224\200â\224\200â\224\200
â\224\220â\224\202
   â\224\202Eventsâ\224\234â\224\200â\226°â\224\202E.S.B.â\224\234â\224\200â\226°
â\224\202 mapping â\224\234â\224\200â\226°â\224\202S.S.B.â\224\234â\224\200â\226°
â\224\202 loader â\224\234â\224\200â\226°â\224\202â\224\202K.G.â\224\202â\224\202
   â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\230  â
\224\202      â\224\202  â\224\202         â\224\202  â\224\202      â\224\202  â
\224\202        â\224\202  â\224\202â\224\224â\224\200â\224\200â\224\200â\224\200
â\224\230â\224\202
            â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224
\230  â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\230  â\224\224â\224\200â\224\200â\224¬â\224\200â\224\200â
\224\200â\224\230  â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â\224
\200â\224\200â\224\200â\224\230  â\224\224â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\230
                                    â\224\202
                  â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224
\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224
\200â\224\200â\224\200â\224\200â\224\200â\224´â\224\200â\224\200â\224\200â\224
\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224
\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224
\200â\224\220
                  â\224\202(event/LOG_login_03)=>(object/RES/router1)â\224\202
                  â\224\224â\224\200â\224\214â\224\200â\224\200â\224\200â\224
\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224
\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224
\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224
\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224
\200â\224\200â\224\200â\224\220
                    â\224\202(event/LOG_login_03)=>(object/RES/router1)â\224\202
                    â\224\224â\224\200â\224\214â\224\200â\224\200â\224\200â\224
\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224
\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224
\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224
\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224
\200â\224\200â\224\200â\224\220
                      â\224\202(event/LOG_login_03)=>(object/RES/router1)â\224
```

\202
â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\230

Figure 8: KG-only data integration architecture for event data streams.

<object/RES_router3>
<object/RES_router2> â\224\202
â\224\202 â\224\202 â\224\214â\224\200â
\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\220
<object/RES_router1>â\224\200rdf:typeâ\224\200â\224¤Resourceâ\224
\202
â\224\202 â\224\224â\224\200â\224\200
â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\230
â\224\202
logOriginatingManagedObject
â\224\202
<event/LOG_login_01> â\224\214â\224\200â\224\200â\224
\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224
\220
<event/LOG_login_02>â\224\200â\224\200rdf:typeâ\224\200â\224¤Ev
entRecordâ\224\202
<event/LOG_login_03> â\224\224â\224\200â\224\200â\224
\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224
\230

Figure 9: Resulting knowledge representation for the KG-only data integration architecture for event data streams

As event streams can be high-paced, it could be beneficial to leverage input/output (I/O) performance optimizations specific to each type of database management system (DBMS), such as Time-Series DataBases (TSDBs) for streaming data and graph databases for knowledge graphs. Figure 10 illustrates the capability to handle both a knowledge graph and a time-series representation of the network's lifecycle while maintaining a link between the two representations (Figure 11). Each serve different purposes, such as context analysis with the knowledge graph representation and trend analysis with the TSDB. Thanks to the linking between the two storage systems, users browsing aggregated data from the knowledge graph can access the raw data within the relevant time span for further analysis, and vice versa.

```
                      â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224
\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\220
                      â\224\202  Complex   â\224\202
                      â\224\202   Event    â\224\202
                      â\224\202 Processing â\224\202
                      â\224\224â\224\200â\224\200â\224\200â\224\200â\224¬â\224\200
â\224\200â\224¬â\224\200â\224\200â\224\200â\224\200â\224\230
             â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224
\220  â\224\214â\224\200â\224\200â\224´â\224\200â\224\200â\224´â\224\200â\224\200
â\224\200â\224\220  â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224
\200â\224\220  â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\220  â\224\214â\224\200â\224\200â\224\200â\224\200â\224
\200â\224\200â\224\220
   â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\220  â
\224\202      â\224\202  â\224\202 Stream  â\224\202  â\224\202      â\224\202  â
\224\202 Stream â\224\202  â\224\202â\224\214â\224\200â\224\200â\224\200â\224\200
â\224\220â\224\202
   â\224\202Eventsâ\224\234â\224\200â\226ºâ\224\202E.S.B.â\224\234â\224\200â\226º
â\224\202 mapping â\224\234â\224\200â\226ºâ\224\202S.S.B.â\224\234â\224\200â\226º
â\224\202 loader â\224\234â\224\200â\226ºâ\224\202â\224\202K.G.â\224\202â\224\202
   â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\230  â
\224\202      â\224\202  â\224\202         â\224\202  â\224\202      â\224\202  â
\224\202        â\224\202  â\224\202â\224\224â\224\200â\224\200â\224\200â\224\200
â\224\230â\224\202
             â\224\224â\224\200â\224\200â\224¬â\224\200â\224\200â\224\200â\224
\230  â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\230  â\224\224â\224\200â\224\200â\224¬â\224\200â\224\200â
\224\200â\224\230  â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â\224
\200â\224\200â\224\200â\224\230  â\224\224â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\230
                 â\224\202                      â\224\202
                 â\224\202  â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200
â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200
â\224\200â\224\200â\224\200â\224\200â\224\200â\224´â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\220
                 â\224\202  â\224\202(event/AIS_login_01)=>(object/RES/router1)â
\224\202
                 â\224\202  â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200
â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200
â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200
â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200
â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200
â\224\200â\224\230
                 â\224\202                             â\224\214â\224\200â\224\200
â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\220  â\224\214â\224
\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\220
```

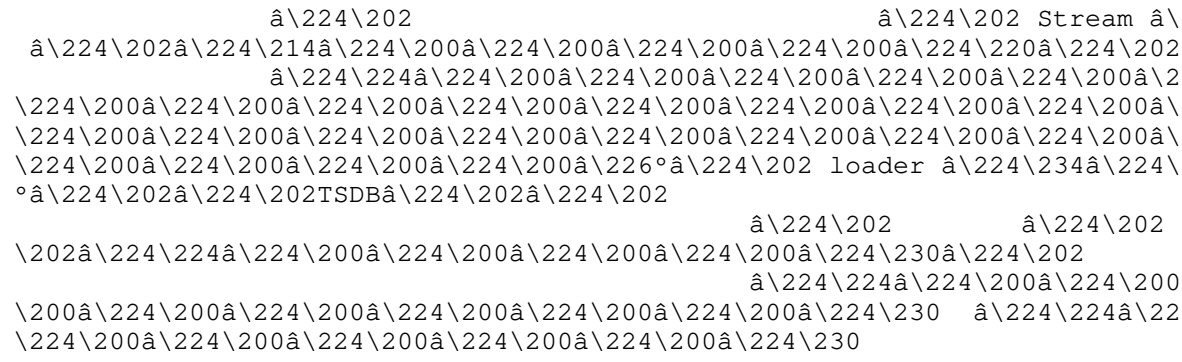


Figure 10: Mixed KG/non-KG data integration architecture for event data streams.

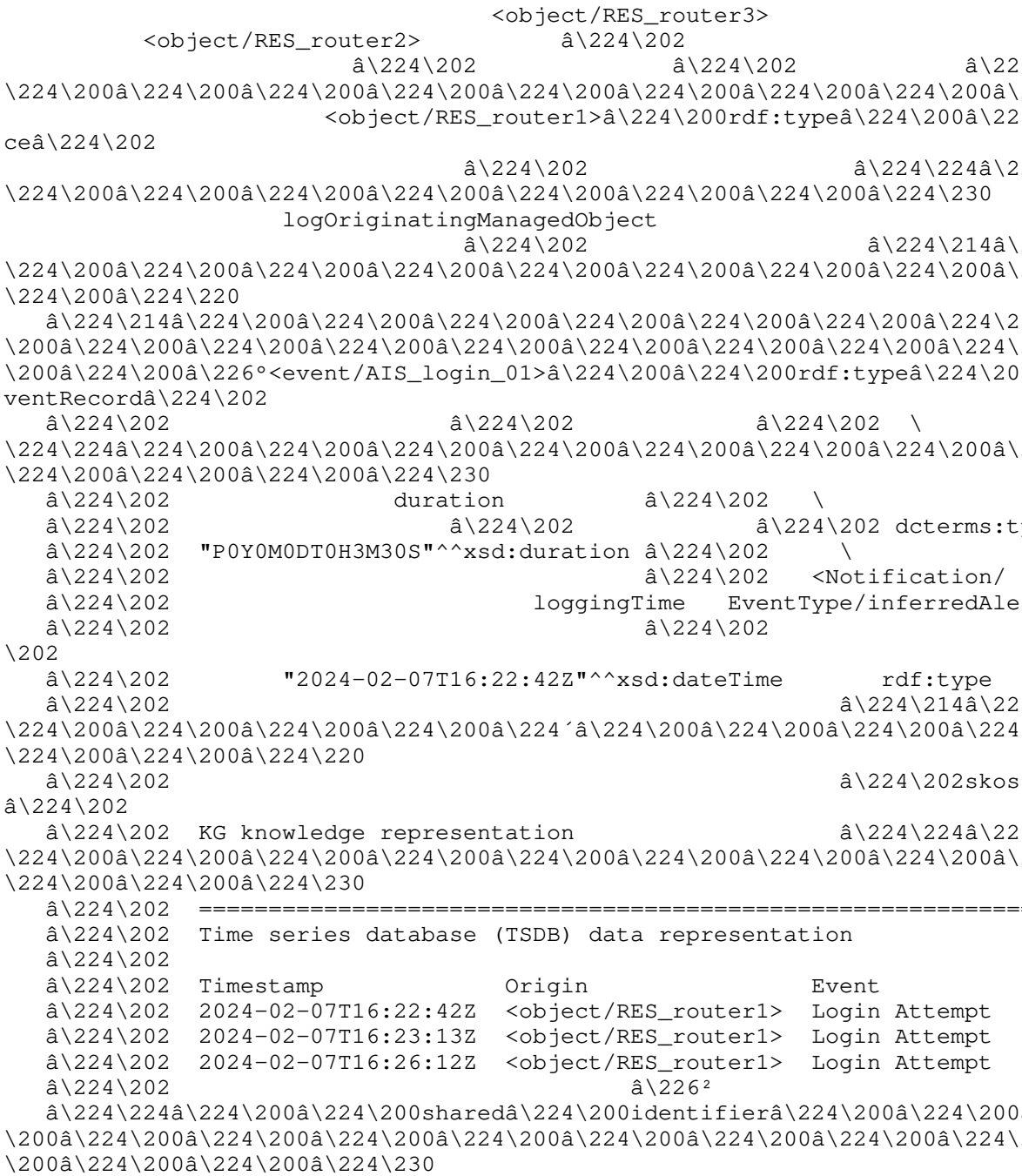


Figure 11: Resulting knowledge representation for the mixed KG/ non-KG data integration architecture for event data streams.

4.3.2. Federated Data Architecture

The Figure 12 illustrates the principles for providing unified access to data distributed across various technological platforms and stakeholders thanks to Federated Queries [SPARQL11-FQ] and the use of a shared ONTO-ITSM across data management platforms.

```
  â\224\200â\224\200â\224\200On-premiseâ\224\200â\224\200â\224\200â\224\200â\224
\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224
\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224
\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200  â\224\214â\224\200â\224\220  S
cope-based querying
  â\224\214Dom.â\224\200Aâ\224\200â\224\220                                    â
\224\202 â\224\202
  â\224\202â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224\220â\224
\202  â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\220
        â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\220  â\224\202 â\224\202           â\224\214â\224\200â\224
\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224
\200â\224\220
â\224\200â\226ºâ\224\202â\224\202 KG  â\224\202â\224\202â\227\204â\224\200â\224¤K
GDBMSâ\224\234â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\200â\224\200â\224¤SPARQL EPâ\224\234â\224\200â\226ºâ\224
\202 â\224\234â\224\200Network &â\224\200â\224¤  NetOps   â\224\202
  â\224\202â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â\224\230â\224
\202  â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\230
        â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\230  â\224\202 â\224\234â\224\200Usageâ\224\200â\224\200â
\224\200â\224\200â\224\200â\224¤Applicationâ\224\202
  â\224\224UG.â\224\2002â\224\200â\224\200â\224\230
    â\224\202 â\224\202           â\224\224â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\230
  â\224\214Dom. Bâ\224\200â\224\220                                 â\224\202 â
\224\202           â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224
\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\220
  â\224\202â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224\220â\224
\202  â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\220
        â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\220  â\224\202 â\224\234â\224\200Network &â\224\200â\224¤
 SecOps   â\224\202
â\224\200â\226ºâ\224\202â\224\202 KG  â\224\202â\224\202â\227\204â\224\200â\224¤K
GDBMSâ\224\234â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\200â\224\200â\224¤SPARQL EPâ\224\234â\224\200â\226ºâ\224
\202 â\224\234â\224\200Securityâ\224\200â\224\200â\224¤Applicationâ\224\202
  â\224\202â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â\224\230â\224
\202  â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\230
        â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\230  â\224\202Fâ\224\202           â\224\224â\224\200â\224
\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224
\200â\224\230
  â\224\224UG.â\224\2001â\224¬â\224\200â\224\230
 â\224\202Eâ\224\202
        â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\202Dâ\224\202â\224\200â\224\200â\224\200â\224\200â\224\200
â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\220
  â\224\200â\224\200â\224\200On-premise / public-cloudâ\224\200â\224\200â\224\200
â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200
â\224\200  â\224\202Eâ\224\202             â\224\202
  â\224\214Dom.â\224\200Câ\224\200â\224\220                                    â
\224\202Râ\224\202             â\226¼  Usage
  â\224\202â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224\220â\224
\202  â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\220 â
\224\214â\224\200â\224\200â\224\200â\224\220     â\224\214â\224\200â\224\200â\224
\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\220  â\224\202Aâ
\224\202           â\224\214â\224\200â\224\200â\224\200â\224\200scopeâ\224\200â
\224\200â\224\220
```

â\224\200â\226°â\224\202â\224\202 RDB â\224\202â\224\202â\227\204â\224\200â\224¤R
DBMS â\224\234â\224\200â\224¤VKGâ\224\234â\224\200â\224\200â\224\200â\224\200â
\224\200â\224¤SPARQL EPâ\224\234â\224\200â\226°â\224\202Tâ\224\202 â
\224\202* â\224\202
â\224\202â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â\224\230â\224
\202 â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\230 â
\224\224â\224\200â\224\200â\224\200â\224\230 â\224\224â\224\200â\224\200â\224
\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\230 â\224\202Eâ
\224\202 Network â\224\202 * * â\224\202
â\224\224UG.â\224\2001&2â\224\230 â\224\202Dâ
\224\202 scopeâ\224\200â\224\200â\224\200â\224\202â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\200â\224\200â\224\200â\224\220 â\224\202
â\224\214Dom.â\224\200Dâ\224\200â\224\220 â
\224\202 â\224\202 â\224\202 â\224\202 * * â\224\202 â\224\202
â\224\202â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224\220â\224
\202 â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\220 â
\224\214â\224\200â\224\200â\224\200â\224\220 â\224\214â\224\200â\224\200â\224
\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\220 â\224\202Qâ
\224\202 â\224\202 *â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200
â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\230
â\224\200â\226°â\224\202â\224\202NoSQLâ\224\202â\224\202â\227\204â\224\200â\224¤R
DBMS â\224\234â\224\200â\224¤VKGâ\224\234â\224\200â\224\200â\224\200â\224\200â
\224\200â\224¤SPARQL EPâ\224\234â\224\200â\226°â\224\202Uâ\224\202 â\224
\202 â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\200â\224\200â\224\220
â\224\202â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â\224\230â\224
\202 â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\230 â
\224\224â\224\200â\224\200â\224\200â\224\230 â\224\224â\224\200â\224\200â\224
\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\230 â\224\202Eâ
\224\202 â\224\202* â\224\202 * * â\224\202 â\224\202
â\224\224UG.â\224\2001â\224\200â\224\200â\224\230
â\224\202Râ\224\202 â\224\224â\224\200â\224\200â\224\202â\224\200â\224
\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\230 â
\224\202
â\224\214Dom.â\224\200Eâ\224\200â\224\220 â
\224\202Iâ\224\202 â\226² â\224\202 * â\224\202
â\224\202â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\220â\224
\202 â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\220 â
\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\220
â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200
â\224\200â\224\220 â\224\202Eâ\224\202 â\224\202 â\224\202 * * â
\224\202
â\224\200â\226°â\224\202â\224\202 LPG â\224\202â\224\202â\227\204â\224\200â\224¤G
DBMS â\224\234â\224\200â\224¤QL tlt.â\224\234â\224\200â\224¤SPARQL EPâ\224\234â
\224\200â\226°â\224\202Sâ\224\202 â\224\202 â\224\224â\224\200â\224\200Sec
urityâ\224\200â\224\230
â\224\202â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â\224\230â\224
\202 â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\230 â
\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\230
â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200
â\224\200â\224\230 â\224\202 â\224\202 â\224\202 scope â\226²
â\224\224UG.â\224¬2â\224\200â\224\200â\224\230
â\224\202 â\224\202 â\224\202 â\224\202
â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\200â\224\200â\224\202 â\224\202â\224\200â\224\200â\224\200
â\224\200â\224\200â\224\200â\224\200â\224\200â\224¼â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\230
â\224\202 â\224\202 â\224\202
â\224\200â\224\200â\224\200Public-cloudâ\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\200â\224\200 â\224\202 â\224\202 â\224\202
â\224\214Dom.â\224\200Fâ\224\200â\224\220 â

\224\202 â\224\202 â\224\202
â\224\202â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\220â\224
\202 â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\220
â\224\214â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\220 â\224\202 â\224\202 â\224\202
â\224\200â\226°â\224\202â\224\202 KG â\224\202â\224\202â\227\204â\224\200â\224¤K
GDBMSâ\224\234â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\200â\224\200â\224¤SPARQL EPâ\224\234â\224\200â\226°â\224
\202 â\224\202 â\224\202
â\224\202â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â\224\230â\224
\202 â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\230
â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\230 â\224\202 â\224\202 â\224\202
â\224\224UG.â\224¬1&2â\224\230 â\224\224â\224
\200â\224\230 â\224\202
â\224\224â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â
\224\200â\224\200â\224\200â\224\200â\224\200â\224\200â\224\230

Figure 12: Unified access to data distributed across various technological platforms.

5. Experiments

5.1. Experimental Plan

In terms of experimentation, we consider the YANG-KG-SEMANTIC-GENERALIZATION case defined in Section 4 as the reference approach and recommend implementing a data processing pipeline that performs the following use cases:

Y-MODEL-FROM-DATA: Based on a dataset of configuration data expressed in YANG models, the goal is to enable extracting the list of models involved for their conversion to their RDFS/OWL equivalent.

Y-MODEL-DEPENDENCIES: Based on a given YANG model, the goal is to enable identifying and retrieving all the YANG models that the model refers to, in order to build a complete corpus of models for their conversion to their RDFS/OWL equivalent as a coherent set.

Y-MODEL-TO-RDFS-OWL: Based on a YANG model and the associated model corpus (i.e. Y-MODEL-DEPENDENCIES), the goal is to enable producing a semantically equivalent RDFS/OWL representation (i.e. ONTO-YANG-MODEL).

Ideally, a YANG to RDFS/OWL/YANG projection algebra would be used to provide a formal proof of semantic equivalence; testing mechanisms should be implemented as a fallback to provide a proof of equivalence.

Y-INSTANCE-TO-KG: Based on a dataset of configuration data expressed in YANG models and the related (set of) ONTO-YANG-MODEL, the goal is to enable constructing a knowledge graph from the configuration data, with the knowledge graph structured by the (set of) ONTO-YANG-MODEL.

Y-MODEL-META-KG-ALIGNMENT: Based on a corpus of YANG models transformed into RDFS/OWL (i.e. Y-MODEL-TO-RDFS-OWL) and a reference ontology structuring the ITSM-KG, the goal is to enable querying of the configuration entities present in the graph (i.e. data derived from the Y-INSTANCE-TO-KG case) through the concepts of the reference ontology.

In addition to identifying the class and property correspondences between the resulting Y-MODEL-TO-RDFS-OWL models and the reference ontology, this capability requires implementing a necessary and sufficient number of class equivalence relations and property equivalence relations.

META-KG-BEHAVIORAL-MODEL: Based on the ITSM-KG, which results from

the composition of the Y-INSTANCE-TO-KG case with Y-MODEL-META-KG-ALIGNMENT and additional operational data structured by ONTO-META, the goal is to learn behavioral models (e.g. incident signatures) in a formalism that can be interpreted through the lenses of ONTO-ITSM and shared with other stakeholders with minimal discrepancies in the underlying configuration data.

5.2. Implementation Status

This section provides pointers to existing open source implementations of this document or in close relation to it.

5.2.1. NORIA

The NORIA project aims at enabling advanced network anomaly detection using knowledge graphs. Among the components resulting from this project, the following ones serve the use case described in this document:

* NORIA-O [NORIA-O-2024], is a data model for IT networks, events and operations information. The ontology is developed using web technologies (e.g. RDF, OWL, SKOS) and is intended as a structure for realizing an ITSM knowledge graph for Anomaly Detection (AD) and Risk Management applications. The NORIA-O implementation is available as open source at https://w3id.org/noria/ (https://w3id.org/noria/). Its use for anomaly detection is discussed in:

  - [SLKG-2023] with a model-based design approach (i.e. query the graph to retrieve anomalies and their context) and a statistical learning approach (i.e. relate entities based on context similarities, then use this relatedness to alert and guide the repair).

  - [GPL-2024] with a process mining approach to align a sequence of entities to activity models, then use this relatedness to guide the repair actions.

  - [NORIA-UI-2024] a Web-based knowledge graph exploration design for incident management that combines the above [SLKG-2023] and [GPL-2024] techniques for broader coverage of anomaly cases and knowledge capitalization.

* A knowledge graph-based platform design [NORIA-DI-2023] using Semantic Web technologies and open source data integration tools to build an ITSM knowledge graph:

- SMASSIF-RML, a Semantic Web stream processing solution with declarative data mapping capability. Available as open source at https://github.com/Orange-OpenSource/smassif-rml (https://github.com/Orange-OpenSource/smassif-rml).

- ssb-consum-up, a Kafka to SPARQL gateway enabling end-to-end Semantic Web data flow architecture with a Semantic Service Bus (SSB) approach. Available as open source at https://github.com/Orange-OpenSource/ssb-consum-up (https://github.com/Orange-OpenSource/ssb-consum-up).

- grlc, a fork of CLARIAH/grlc with SPARQL UPDATE and GitLab interface features to facilitate the call and versioning of stored user queries in SPARQL syntax (e.g. for anomaly detection following the model-based design approach). Available as open source at https://github.com/Orange-OpenSource/grlc (https://github.com/Orange-OpenSource/grlc).

* SemNIDS [SemNIDS-2023], a test bench involving network trafic generation, open source Network Intrusion Detection Systems (NIDS), knowledge graphs, process mining and conformance checking components.

Note that the NORIA project does not currently address the Y-MODEL-FROM-DATA, Y-MODEL-DEPENDENCIES, and Y-MODEL-TO-RDFS-OWL use cases.

5.2.2. YANG2OWL

The YANG2OWL framework aims at facilitating the implementation of a Network Digital Twin (NDT) that would leverage the representation and reasoning capabilities typically associated with knowledge graphs for anomaly detection needs, as well as for network management purposes by enabling network configuration based on modifications at the level of the ITSM-KG itself. Basically, the approach consists of reusing YANG data models used in network operations in a nearly equivalent form within Semantic Web technologies (i.e. producing ONTO-YANG-MODEL instances) to create a bijection between network configuration data and the NDT.

The YANG2OWL framework addresses the use cases Y-MODEL-TO-RDFS-OWL and Y-INSTANCE-TO-KG (as defined in Section 5.1).

Figure 13 illustrates the top-level tasks of the semantization process at play. Subsequent sections detail how the framework builds ontologies that captures the specificities of the telco domain and models any telco network instance as an ITSM-KG. Please note that the publication of the related tools and algorithms is in progress.

Figure 13: The YANG2OWL framework. Labels within boxes represent automated or human actions, while labels between top/bottom lines represent datasets

5.2.2.1. Motivations and Principles

The document [I-D.mackey-nmop-kg-for-netops] (Knowledge Graph Framework for Network Operations) emphasizes the importance of ontologies alongside knowledge graphs for network management automation. However, it lacks guidance on creating these ontologies and provides limited details on generating knowledge graphs or their relationship with the ontologies. To address these topics, the following principles have been considered to underpin the development of the YANG2OWL approach:

1. The ontologies should intimately reflect YANG models,

2. The generation of ontologies should be mostly automatized,

3. The knowledge graphs should intimately reflect the payload of messages that YANG compliant network equipments and controlers publish or emit in response to a Remote Procedure Call (RPC) request,

4. The generation of knowledge graphs should be automated,

5. The nodes and predicates of the knowledge graphs should be defined as instances of classes and properties of the ontologies.

Point 1 of the proposed principles is essential for ensuring the engagement of network administrators and experts in semantic technology. Aligning the ontology's vocabulary (class and relationship naming) and semantics (relationship constraints) with that of network managers is crucial. The YANG language is currently the reference in this area and will continue to be so, given its specification by the IETF and support from major telco industry players. This necessity has driven the development of the YANG2OWL framework for converting YANG models into OWL models, which corresponds to point 2 of the proposal. Points 3, 4, and 5 are direct outcomes of the commitment to points 1 and 2.

5.2.2.2. The Y-MODEL-TO-RDFS-OWL step

YANG and OWL are both data modeling languages. They define a vocabulary and a grammar. The vocabulary defines the concept of the domain. YANG domain is the telco domain.

In a natural language, the vocabulary defines nouns, verbs, adjectives, and adverbs that are useful for discussing the world. The grammar specifies how these elements should be assembled into sentences that describe a state of the world. In a YANG model, the vocabulary is defined in terms of _containers_, _lists_, _leaves_, _leaf-lists_, and other categories, while the grammar is defined in terms of statements that relate these elements to one another. In an OWL ontology, the vocabulary is defined in terms of _classes_, _subclasses_, _object properties_, and _data properties_, which is somewhat similar to YANG but does not directly map.

As ontologies have been introduced as a modeling language meant to share a common view (or knowledge) of a domain among different stakeholders [GRUBER-1995], the terms defined by the ontologies should reflect those used by equipment manufacturers, telecom solutions developers, systems integrators, network operators, and ultimately end users.

A YANG model is a document containing declarations. The document has a tree-like structure: declarations can contain other declarations. There are about half hundred types of declarations. The main ones are _container_, _list_, _leaf_ and _leaf-list_:

CONTAINER: It is a concept, something we can talk about ; it is the the basic type of elements of the domain, such as a network, a node, a link. A container declaration can contain another container declaration that can be called a sub-container. This

sub-container allows to define a concept that will characterize the container that contains it (e.g. link, source, and destination).

LIST: It is a concept that can have multiple instances, such as nodes of a network.

LEAF: It is a property of this concept, such as an identifier or a geographical location.

LEAF-LIST: It is a multivalued property, such as hours of the day the device is in sleep mode.

By applying the above principles, and in line with the reasons sketched in Section 5.2.2.1, we have developed the YANG2OWL that automatically generates OWL ontologies from YANG modules (i.e. computes ONTO-YANG-MODELs). Figure 14 sketches the use of the YANG2OWL tool to compute the org.opendaylight.yangtools ONTO-YANG-MODEL.

Figure 14: Computing the org.opendaylight.yangtools ONTO-YANG-MODEL with YANG2OWL.

In more detail, we have defined mapping rules between YANG constructs and OWL concepts and implemented these in YANG2OWL. The main YANG constructs (_container_, _list_, _leaf_, and _leaf-list_) are transformed as follows:

* The _container_ and _list_ declarations are converted into OWL classes. The name of the OWL class correponds to the name of the _container_ or _list_ in the YANG model.

* The _leaf_ and _leaf-list_ declarations are converted into OWL data properties. The name of the OWL data property corresponds the name of the _leaf_ or _leaf-list_ in the YANG model.

An example of this conversion is presented in the following section.

5.2.2.3. The Y-INSTANCE-TO-KG step

As introduced above, YANG models define the vocabulary and grammar to describe factual knowledge about the state of the network. For example if a YANG module defines the container _node_, and this container has a leaf identifier which has the type string, then a valid JSON document with configuration data describing a node should be a JSON object containing a key named identifier which value should be a string such as router_253.

So, in line with the mapping rules of YANG statement into OWL concepts defined in Section 5.2.2.2, when parsing a JSON tree that comply to a given YANG model we can assume that if we get a _key which value is a JSON object_ then the _key should be the name of a container or a list_ and its _value should be a description to be further analyzed_. Thus, in terms of knowledge graph modeling, this JSON object should be interpreted as an _instance of a class_ which name is the _name of the container or of the list_.

Conversely, if the value is a _litteral_, the _key_ should be the _name of a leaf or a leaf-list_. Thus, in terms of knowledge graph modeling, the litteral should be interpreted as the _object of a DataProperty_ which name is the _name of the leaf_.

The JSON2RDF tool (which is part of the YANG2OWL framework) implements these principles, realizing the Y-INSTANCE-TO-KG use case. Figure 15 shows the algorithm implemented by JSON2RDF as pseudo code.

```
function createURI(jsonObject, class, namespace, ontology) {
  if class has a 'key' annotation {
    get the content <keycontent> of this annotation
    search the key <keycontent> in the jsonObject
    append the key to the namespace to create the URI
  } else {
    generate a unique URI
  }
  return the URI created
}

function createObject(URI, class) {
  return an instance of the class with the given URI
}

function parse(object, parentURI, class, namespace, ontology) {
  objectURI = createURI(object,class, namespace, ontology)
  createObject(objectURI, class)
  for each key of object {
    if the value of object[key] is a list {
      for each elt of the list {
        if elt is an object {
          parse(elt, objectURI, key, namespace, ontology)
          create the triple <objectURI haskey elt>
        } else if elt is a literal
            create the triple <objectURI key elt>
    } else if the value of object[key] is an object {
        eltURI = createURI(elt,key, namespace, ontology)
        create the triple <objectURI haskey eltURI>
        parse(elt, objectURI, key, namespace, ontology)
    } else if the value of object[key] is literal {
        create the triple <objectURI key value>
    }
  }
}
```

Figure 15: Pseudo code of the algorithm implemented by JSON2RDF.

The algorithm is initiated by calling the parse function as follows, where top is the root of the JSON object (i.e. configuration data as a JSON tree that complies to a given YANG model), and ontology is the output of the Y-MODEL-TO-RDFS-OWL step:

```
call parse(top, nil, namespace, ontology)
```

5.2.2.4. Example of Implementation

To illustrate the YANG2OWL approach, this section briefly reports on an experiment conducted in an industrial setting with data from a virtualized 5G infrastructure. In the context of the Network Change Management process, _impact analysis_ prior to conducting a scheduled operation can be run on an ITSM-KG. It aims to determine all the components of the 5G core network that are dependent of a given (set of) network infrastructure element. For example, for a scheduled operation on a leaf node (i.e. a network element in a 2-tier spine-leaf architecture), the impact calculus will return all the servers connected to the leaf, all the Virtual Machines (VMs) hosted on these servers, all the Network Functions (NFs) deployed on these VMs, and ideally all the telecom services using these NFs.

Figure 16 provides an overview of the data processing workflow used for the experiment. The tasks of the diagram are described below.

Figure 16: Flowchart for the YANG2OWL experiment. A left vertical bar on a step indicates that it is scripted; otherwise, steps require user or operator action.

Model Gathering: This task corresponds to the realization of the Y-MODEL-FROM-DATA use case with the manual selection of YANG modules in relation to the 3GPP application domain. The YANG modules from [ETSI-TS-128-541] have been selected for this experiment.

Model Translation: For a given YANG module, this task implements the Y-MODEL-DEPENDENCIES use case by fetching sub-YANG modules from well-known GitHub repositories used for storing YANG modules (e.g. IETF, IEEE, IANA, ETSI, broadband forum, OpenROADM, OpenConfig, Cisco, Huawei, to name a few). This is achieved by scrutinizing import clauses (including imports of imports) and examining module locations and relationships from the [YANG-CATALOG]. Additionally, it addresses the Y-MODEL-TO-RDFS-OWL use case using the YANG2OWL solution defined in Section 5.2.2.2. For this experiment, the resulting ontology is referred to as MOBILE-O.

Model Curation: This task involves providing a streamlined ontology by manually _filtering_ (selection of classes and relationships based on the data available) and _grouping_ (compression of the model hierarchy, i.e. class of classes) the model resulting from the _Model Translation_ task. This simplification aims to enhance the readability of the model for an operator and facilitate the implementation of potentially more concise queries in the downstream _Use Cases-Related Querying_ task.

Model-Related Knowledge Graph Construction: It realizes the Y-INSTANCE-TO-KG use case using the JSON2RDF solution described in Section 5.2.2.3.

NetOps-Related Knowledge Graph Construction: It corresponds to the execution of RML transformation rules [RML] with definitions from the NORIA-O ontology [NORIA-O-2024] for the integration of complementary data to that of the 5G network derived from YANG configurations (i.e. the _Model-Related Knowledge Graph Construction_ task), such as the topology of connected networks, scheduled operations, incident tickets, and organization-related data.

Global Knowledge Graph Construction: It is achieved through parallel insertions into a graph database of the results from the _Model-Related_ and _NetOps-Related_ tasks, after ensuring that: 1) the URI patterns implemented in the RML rules of the NetOps-Related step are consistent with the URIs produced by the Model-Related step to benefit from automatic linking of triples within the graph database through the uniqueness of the URIs; 2) the definition of mappings between MOBILE-O and NORIA-O has been implemented and inserted into the graph database (i.e. realization of the Y-MODEL-META-KG-ALIGNMENT use case through the implementation of the ONTO-LINKER concept as illustrated in Figure 5). For this experiment, the graph database is a Neo4j database [NEO4J] instance, and the loading is performed using the Neo4j Neosemantics toolkit.

Use Cases-Related Pre-Processing: Dependency relationships are, in

general, knowledge elements that cannot be directly derived from field data; they are part of the business knowledge regarding the operation of the network systems. It may therefore be beneficial to support the downstream _Use Cases-Related Querying_ task by performing pre-processing, particularly by calculating these dependency relationships retrospectively from business rules and the data loaded into the database. For example, one can create a (Server)-[DEPENDS_ON]->(Leaf) relationship by searching instances of the (Server)-(Server Interface)-(Network Link)-(Leaf Interface)-(Leaf) graph pattern. The same principle can apply to different network configurations to create other kinds of dependency relationships.

For this experiment, the dependency relationships are calculated directly in the graph database using Neo4j Cypher language queries, or externally to the graph database using SHACL shapes [SHACL] according to the principles described in [GUITTOUM-2023]. As another example, more specific to the 3GPP models [ETSI-TS-128-541] included in MOBILE-O and the Neo4j setup, one could calculate a dependency relationship between a 5G NF and the Kubernetes cluster that hosts it, as shown in Figure 17. It is important to note that subclass inference with Neo4j is not automatic and must be performed through dedicated queries, as illustrated in Figure 18.

```
MATCH (c:ManagedFunction)--(n:namespace)--(k:ClusterKubernetes)
MERGE (c)-[d:DEPENDS_ON]->(k)
```

Figure 17: Dependency calculation query, in Cypher syntax, for relating a 5G NF and the Kubernetes cluster that hosts it.

```
MATCH (m)<-[:subClassOf]-(x)<-[:type]-(c)
WHERE m.uri CONTAINS 'ManagedFunction'
SET c:ManagedFunction
```

Figure 18: Subclass inference query, in Cypher syntax, to tag 5G NF entities as ‘ManagedFunction‘ based on prior annotation of the entities at creation time with a specific class described in the YANG model, which is also a subclass of ‘ManagedFunction‘ as per MOBILE-O.

Use Cases-Related Querying: The exploitation of dependency relationships is carried out through queries on the graph, e.g. during the insertion of an entity of type noria:ChangeRequest or by following an exploratory approach by coupling a query such as that in Figure 19 with a visualization tool like Neo4j NeoDash.

```
MATCH (e1) WHERE e1.resourceHostName = $neodash_ressource_hostname
MATCH q1 = (e1) ((w)<-[:DEPENDS_ON]-(t)) {0,8}
UNWIND t AS impacts
RETURN DISTINCT impacts.resourceHostName
```

Figure 19: User query, in Cypher syntax using a quantified path pattern, for rendering dependency relationships in a Neo4j NeoDash display. The query seeks paths starting from the node 'e1' and propagates up to 8 times using the 'DEPENDS_ON' relationships. The depth of 8 has been defined in relation to the characteristics of the networks addressed in the experimentation.

Situation Analysis: Decision-making based on the results of the upstream task is the responsibility of the network administrator, potentially supported by a complementary exploration of the ITSM-KG performed algorithmically or interactively to analyze a broader technical and operational context.

5.2.2.5. Discussion

While the YANG2OWL approach has proven its validity as a proof of concept, several R&D questions remain for exploration with the NMOP community, including:

* Are the conversion principles based on statement types (class vs. data property) in the Y-MODEL-TO-RDFS-OWL use case universally applicable?

* How to ensure that an ITSM-KG can still be generically constructed from JSON/YANG data and queried when a _Model Curation_ task is applied on an ONTO-YANG-MODEL?

* What techniques can automate the Y-MODEL-META-KG-ALIGNMENT use case?

* What principles should guide the implementation of the Y-MODEL-META-KG-ALIGNMENT use case to extract an aggregated view from ONTO-META of infrastructures/configurations represented by an ONTO-YANG-MODEL (e.g. distinguishing devices from sub-devices)?

* As evoked in [I-D.boucadair-nmop-rfc3535-20years-later] (NEW-OPS-REQ-QUICK-BUT-WELL), how can we ensure reliable retrieval of dependencies between YANG modules for the Y-MODEL-DEPENDENCIES use case? Indeed, while browsing the GitHub projects of module developers, we observe a lack of uniformity in the way modules are presented and managed (e.g. differences in project structure, replication and local modifications of reference modules), which hinders dependency calculation and the sound inclusion of sub-modules in the YANG2OWL translation process.

Furthermore, it is noteworthy that the YANG2OWL approach is complementary to the YANG2RDF approach [YANG2RDF-IETF-121], which consists in translating YANG models into RDF. More specifically, YANG2RDF defines an ontology of the YANG language, where RDF graph instances model a YANG module. This approach is useful for querying YANG models. In contrast, the YANG2OWL approach defines an ontology of a YANG model, where RDF graph instances model an operational network. Future work may aim to combine the YANG2RDF and YANG2OWL approaches.

Finally, it is noteworthy that the YANG2OWL framework automates the _Ontology Implementation_ and _Ontology Update_ activities of the LOT4KG methodology [LOT4KG-2024] (a methodology that extends the well-known LOT ontology engineering methodology to include knowledge graph lifecycle management) by linking YANG modules with ITSM-KG fragment construction. This streamlines the development of NDT architectures based on knowledge graphs and simplifies ITSM-KG updates when YANG modules change.

6. Security Considerations

As this document covers the _ITSM-KG_ concepts, and use cases, there is no specific security considerations.

However, as the concept of a meta-knowledge graph involves the construction of a multi-faceted graph (i.e. including network topologies, operational data, and service and client data), it poses the risk of simplifying access to network operational data and functions that fall outside the knowledge graph users' responsibility or that could facilitate the intervention of malicious individuals. To support the discussion on mitigating this risk, we suggest referring to Figure 12, which illustrates the concept of partial access to the meta-knowledge graph based on rights associated with each user group (UG) at the data domain level. We also recommend referring to [AMO-2012] for an example of implementation of access rights in a content management system that relies on Semantic Web models and technologies. This implementation uses the AMO ontology, which includes a set of classes and properties for annotating resources that require access control, as well as a base of inference rules that model the access management strategy to carry out.

7. IANA Considerations

This document has no IANA actions.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/rfc/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/rfc/rfc8174>.

[RFC8345] Clemm, A., Medved, J., Varga, R., Bahadur, N., Ananthakrishnan, H., and X. Liu, "A YANG Data Model for Network Topologies", RFC 8345, DOI 10.17487/RFC8345, March 2018, <https://www.rfc-editor.org/rfc/rfc8345>.

[RFC9418] Claise, B., Quilbeuf, J., Lucente, P., Fasano, P., and T. Arumugam, "A YANG Data Model for Service Assurance", RFC 9418, DOI 10.17487/RFC9418, July 2023, <https://www.rfc-editor.org/rfc/rfc9418>.

8.2. Informative References

[AMO-2012] Buffa, M. and C. Faron-Zucker, "Ontology-Based Access Rights Management", 2012, <https://doi.org/10.1007/978-3-642-25838-1_3>.

[DevOpsInfra-2021] Corcho, O., Chaves-Fraga, D., Toledo, J., Arenas-Guerrero, J., Badenes-Olmedo, C., Wang, M., Peng, H., Burrett, N., Mora, J., and P. Zhang, "A High-Level Ontology Network for ICT Infrastructures", 2021, <https://doi.org/10.1007/978-3-030-88361-4_26>.

[ETSI-TS-128-541] ETSI, "5G; Management and orchestration; 5G Network Resource Model (NRM); Stage 2 and stage 3 (3GPP TS 28.541 version 18.9.0 Release 18)", October 2024, <https://www.etsi.org/deliver/etsi_ts/128500_128599/128541/18.09.00_60/ts_128541v180900p.pdf>.

[FLAGSM-2021] Steenwinckel, B., Paepe, D. D., Hautte, S. V., Heyvaert, P., Bentefrit, M., Moens, P., Dimou, A., Bossche, B. V. D., Turck, F. D., Hoecke, S. V., and F. Ongenae, "FLAGS: A Methodology for Adaptive Anomaly Detection and Root Cause Analysis on Sensor Data Streams by Fusing Expert Knowledge with Machine Learning", 2021, <https://doi.org/10.1016/j.future.2020.10.015>.

[FOLIO-2018] Steenwinckel, B., Heyvaert, P., Paepe, D. D., Janssens, O., Hautte, S. V., Dimou, A., Turck, F. D., Hoecke, S. V., and F. Ongenae, "Towards Adaptive Anomaly Detection and Root Cause Analysis by Automated Extraction of Knowledge from Risk Analyses", 2018, <https://www.ceur-ws.org/Vol-2213/paper2.pdf>.

[GPL-2024] Tailhardat, L., Stach, B., Chabot, Y., and R. Troncy, "Graphameleon: Relational Learning and Anomaly Detection on Web Navigation Traces Captured as Knowledge Graphs", 2024, <https://doi.org/10.1145/3589335.3651447>.

[GRUBER-1995] R., G. T., "Toward principles for the design of ontologies used for knowledge sharing?", 1995, <https://doi.org/10.1006/ijhc.1995.1081>.

[GUITTOUM-2023]
Amal, G., Francois, A., SÃ©bastien, B., Fabienne, B., and D. P. Noel, "Inferring Threatening IoT Dependencies Using Semantic Digital Twins Toward Collaborative IoT Device Management", 2023, <https://doi.org/10.1145/3555776.3578573>.

[I-D.boucadair-nmop-rfc3535-20years-later]
Boucadair, M., Contreras, L. M., de Dios, O. G., Graf, T., Rahman, R., and L. Tailhardat, "RFC 3535, 20 Years Later: An Update of Operators Requirements on Network Management Protocols and Modelling", Work in Progress, Internet-Draft, draft-boucadair-nmop-rfc3535-20years-later-08, 12 May 2025, <https://datatracker.ietf.org/doc/html/draft-boucadair-nmop-rfc3535-20years-later-08>.

[I-D.havel-nmop-digital-map-concept]
Havel, O., Claise, B., de Dios, O. G., and T. Graf, "Digital Map: Concept, Requirements, and Use Cases", Work in Progress, Internet-Draft, draft-havel-nmop-digital-map-concept-00, 4 July 2024, <https://datatracker.ietf.org/doc/html/draft-havel-nmop-digital-map-concept-00>.

[I-D.irtf-nmrg-network-digital-twin-arch]
Zhou, C., Yang, H., Duan, X., Lopez, D., Pastor, A., Wu, Q., Boucadair, M., and C. Jacquenet, "Network Digital Twin: Concepts and Reference Architecture", Work in Progress, Internet-Draft, draft-irtf-nmrg-network-digital-twin-arch-10, 28 February 2025, <https://datatracker.ietf.org/doc/html/draft-irtf-nmrg-network-digital-twin-arch-10>.

[I-D.mackey-nmop-kg-for-netops]
Mackey, M., Claise, B., Graf, T., Keller, H., Voyer, D., Lucente, P., and I. D. Martinez-Casanueva, "Knowledge Graph Framework for Network Operations", Work in Progress, Internet-Draft, draft-mackey-nmop-kg-for-netops-02, 4 March 2025, <https://datatracker.ietf.org/doc/html/draft-mackey-nmop-kg-for-netops-02>.

[I-D.marcas-nmop-knowledge-graph-yang]
Martinez-Casanueva, I. D., RodrÃguez, L. C., and P. Martinez-Julia, "Knowledge Graphs for YANG-based Network Management", Work in Progress, Internet-Draft, draft-marcas-nmop-knowledge-graph-yang-05, 21 October 2024, <https://datatracker.ietf.org/doc/html/draft-marcas-nmop-knowledge-graph-yang-05>.

[I-D.netana-nmop-network-anomaly-lifecycle]
Riccobene, V., Roberto, A., Graf, T., Du, W., and A. H. Feng, "An Experiment: Network Anomaly Lifecycle", Work in Progress, Internet-Draft, draft-netana-nmop-network-anomaly-lifecycle-05, 3 November 2024, <https://datatracker.ietf.org/doc/html/draft-netana-nmop-network-anomaly-lifecycle-05>.

[LOT4KG-2024]
Romana, P., MarÃ-a, P.-V., Diego, C.-H., David, C.-F., and S. Lise, "When Ontologies Met Knowledge Graphs: Tale of a Methodology", 2024, <https://doi.org/10.1007/978-3-031-78952-6_43>.

[NEO4J]
Neo4j, Inc., "Neo4j - Graph Database & Analytics", n.d., <https://neo4j.com/>.

[NORIA-DI-2023]
Tailhardat, L., Troncy, R., and Y. Chabot, "Designing NORIA: a Knowledge Graph-based Platform for Anomaly Detection and Incident Management in ICT Systems", 2023, <https://ceur-ws.org/Vol-3471/paper3.pdf>.

[NORIA-O-2024]
Tailhardat, L., Troncy, R., and Y. Chabot, "NORIA-O: An Ontology for Anomaly Detection and Incident Management in ICT Systems", 2024, <https://doi.org/10.1007/978-3-031-60635-9_2>.

[NORIA-UI-2024]
Tailhardat, L., Chabot, Y., Py, A., and P. Guillemette, "NORIA UI: Efficient Incident Management on Large-Scale ICT Systems Represented as Knowledge Graphs", 2024, <https://doi.org/10.1145/3664476.3670438>.

[ONTO-MATCH-2022]
Jan, P., Guilherme, C., Karolin, S., Katharina, K., Michael, H., and P. Heiko, "Ontology Matching Through Absolute Orientation of Embedding Spaces", 2022, <https://doi.org/10.1007/978-3-031-11609-4_29>.

[OWL]
W3C, "OWL 2 Web Ontology Language Document Overview (Second Edition)", December 2012, <https://www.w3.org/TR/owl2-overview/>.

[RDF]
W3C, "Resource Description Framework (RDF): Concepts and Abstract Syntax", February 2014, <https://www.w3.org/TR/rdf11-concepts/>.

[RDFS] W3C, "RDF Schema 1.1", February 2014, <https://www.w3.org/TR/rdf-schema/>.

[RML] Dimou, A., Sande, M. V., Meester, B. D., Heyvaert, P., and T. Delva, "RDF Mappling Language (RML)", June 2024, <https://rml.io/specs/rml/>.

[SemNIDS-2023] Ferrero, D., Agarwalla, Y., Tailhardat, L., and T. Ehrhart, "SemNIDS, bringing semantics into Network Intrusion Detection Systems", 2023, <https://github.com/D2KLab/SemNIDS>.

[SHACL] W3C, "Shapes Constraint Language (SHACL)", July 2017, <https://www.w3.org/TR/shacl/>.

[SKOS] W3C, "SKOS Simple Knowledge Organization System Reference", August 2009, <https://www.w3.org/TR/skos-reference/>.

[SLKG-2023] Tailhardat, L., Troncy, R., and Y. Chabot, "Leveraging Knowledge Graphs For Classifying Incident Situations in ICT Systems", 2023, <https://doi.org/10.1145/3600160.3604991>.

[SPARQL11-FQ] W3C, "SPARQL 1.1 Federated Query", March 2013, <https://www.w3.org/TR/sparql11-federated-query/>.

[SPARQL11-QL] W3C, "SPARQL 1.1 Query Language", March 2013, <https://www.w3.org/TR/sparql11-query/>.

[YANG-CATALOG] Cisco and IETF, "YANG Catalog", n.d., <https://www.yangcatalog.org/>.

[YANG2RDF-IETF-121] Michael, M., Anatolii, P., and C. Benoit, "YANG 2 RDF", 2024, <https://datatracker.ietf.org/doc/slides-121-nmop-yang-2-rdf/>.

Acknowledgments

We would like to thank Benoit Claise for spontaneously seeking to include the work of the NORIA research project in the vision of the NMOP working group through direct contact. We also extend our gratitude to Mohamed Boucadair for facilitating discussions within the NMOP community and for providing advice in organizing this Internet Draft.

Additionally, we would like to thank Fano Ramparany for his initial analysis of the possibilities of defining a model conversion algebra for going from YANG data models to OWL ontologies (draft-tailhardat-nmop-incident-management-noria-01).

Changes Between Revisions

v00 - v01 (draft-tailhardat-nmop-incident-management-noria)

* Added details to the _An ITSM-KG for Learning and Sharing Network Behavioral Models_ section (formerly called _A meta-knowledge graph to align operator-specificities and share behavioral models of technical architectures_).

* Added the Experiments / NORIA approach.

v01 - v02

* Added the Experiments / YANG2OWL framework based on details from Fano RAMPARANY (Orange Research), Pauline FOLZ (Orange Research), and Fabrice BLACHE (Orange Research).

* Added the Experiments / YANG2OWL example based on details from Romain VINEL (Orange France), ClÃ©ment GOUILLOUD (SOFRECOM), Arij ELMAJED (Orange France), and Lionel TAILHARDAT (Orange Research).

Contributors

Maria Massri
Orange Research
Email: maria.massri@orange.com

Fabrice Blache
Orange Research
Email: fabrice.blache@orange.com

SÃ©bastien Bolle
Orange Research
Email: sebastien.bolle@orange.com

Thomas Hassan
Orange Research
Email: thomas.hassan@orange.com

Romain Vinel
Orange France
Email: romain.vinel@orange.com

Arij Elmajed
Orange France
Email: arij.elmajed@orange.com

ClÃ©ment Gouilloud
SOFRECOM
Email: clement.gouilloud@sofrecom.com

Authors' Addresses

Lionel Tailhardat
Orange Research
Email: lionel.tailhardat@orange.com

RaphaÃ«l Troncy
EURECOM
Email: raphael.troncy@eurecom.fr

Yoan Chabot
Orange Research
Email: yoan.chabot@orange.com

Fano Ramparany
Orange Research
Email: fano.ramparany@orange.com

Pauline Folz
Orange Research
Email: pauline.folz@orange.com