Network Management Research Group
Internet-Draft
Intended status: Informational
Expires: 8 January 2026

M. Gu, Ed.
J. Jeong, Ed.
Y. Ahn
Sungkyunkwan University

7 July 2025

An Intent Translation Framework for IoT Networks
draft-gu-nmrg-intent-translator-01

Abstract

The evolution of 6G networks and the expansion of Internet of Things (IoT) environments introduce new challenges in managing diverse networked resources. Intent-based management frameworks enable operators to express desired network outcomes using high-level intents, often articulated in natural language. However, converting these expressions into machine-executable policy configurations remains an open challenge.

This document defines an intent translation framework designed to bridge the gap between user-issued intents and structured policy representations for 6G enabled IoT systems. The framework accepts natural language intent as input and produces a policy document in a structured format, such as YAML, that aligns with the intent model defined in 3GPP in [TS-28.312].

The framework consists of modular components responsible for processing input, aligning user intent with domain knowledge, evaluating semantic confidence, and generating standardized output. This modularity supports transparency, interoperability, and automated policy enforcement in next-generation network infrastructures.

This Internet-Draft will expire on 8 January 2026.

Table of Contents

1. Introduction

The rapid growth of Internet of Things (IoT) deployments and the evolution toward 6G networks have introduced increasing complexity in the management of heterogeneous devices, services, and application policies. As operational environments scale, the traditional model of manually configuring service-level policies becomes unsustainable.

Intent-based management is a paradigm that allows administrators to specify desired outcomes through high-level intents, often expressed in natural language. These intents must then be interpreted, validated, and translated into structured policy representations that can be executed by network functions and orchestrators. While

standards such as 3GPP TS 28.312 define lifecycle procedures for managing intents in mobile networks, they do not specify mechanisms for interpreting natural language or translating it into compliant policy structures.

This document defines a modular intent translation framework that addresses this gap. The framework enables automated conversion of user-issued intents into structured policy outputs in formats such as YAML, aligned with the expectations and procedures defined in 3GPP TS 28.312. It supports a range of use cases across IoT and 6G domains, including resource optimization, security management, and service quality assurance.

The framework is composed of functional components that operate sequentially or in coordination:

* *Intent Processing Component:*Accepts and interprets user-provided intents into structured representations.

* *Semantic Alignment Component:*Matches the processed intent to relevant domain knowledge for policy resolution.

* *Confidence Evaluation Component:*Assesses interpretation reliability and identifies degraded or low-confidence mappings.

* *Policy Generation Component:*Produces a machine-readable policy in a structured format suitable for deployment.

The design promotes modularity, transparency, and alignment with existing network automation architectures. It enables consistent translation of operator goals into policies that are interoperable with standard orchestration and management systems in 6G-enabled IoT networks.

2. Terminology

This document uses the terminology defined in [RFC9315], [RFC8329], [I-D.ietf-i2nsf-applicability], [I-D.jeong-i2nsf-security-management-automation], and [I-D.yang-i2nsf-security-policy-translation].

In addition, the following terms are defined for the purpose of this document:

* *Intent:* A set of operational goals (that a network should meet) and outcomes (that a network is supposed to deliver), defined in a declarative manner without specifying how to achieve or implement them [RFC9315].

* *Intent-Based Management (IBM):* It enforces an intent from a user (or administrator) into a target system. An intent can be expressed as a natural language sentence and translated into a high-level policy using natural language interpretation and structured policy mapping [I-D.jeong-i2nsf-security-management-automation], [I-D.yang-i2nsf-security-policy-translation]. This high-level policy is then transformed into low-level policies that are dispatched to appropriate Service Functions (SFs). Based on monitoring feedback, new rules may be generated or existing policies refined.

* *Intent Processing Component:* A logical function that receives a natural language input from the user or operator and produces a structured internal representation of the intent. This component facilitates the initial abstraction required for intent lifecycle operations [RFC9315], [I-D.jeong-i2nsf-security-management-automation].

* *Semantic Alignment Component:* A logical function that interprets the structured intent and determines its best alignment with existing domain knowledge or policy databases. The purpose of this component is to ensure the intent maps to a resolvable and enforceable policy outcome.

* *Confidence Evaluation Component:* A logical function that estimates the reliability or confidence of semantic intent translation. Low-confidence outputs may be flagged as degraded and subjected to fallback, verification, or reprocessing.

* *Degraded Intent:* An intent translation result that has been marked as low-confidence due to weak alignment, missing information, or uncertainty in the reasoning process. A degraded intent may still result in policy generation, but with warnings or limited scope.

* *Policy Generation Component:* A logical function that produces a machine-readable policy, typically in a format such as YAML or JSON, based on the resolved intent and domain mappings. This component ensures compliance with policy schema requirements, such as those defined in 3GPP [TS-28.312].

3. Intent Translator Framework Architecture

This section defines the architecture of the Intent Translator Framework, which is designed to convert high-level, natural language intents into machine-readable policy representations in structured formats such as YAML. The framework enables intent-based management automation for 6G-enabled IoT environments and aligns with policy modeling and lifecycle procedures defined in [TS-28.312].

3.1. Intent Translator

The Intent Translator is a modular subsystem responsible for converting natural language intent into a structured, machine-readable policy document suitable for enforcement in intent-based management systems. It forms the core of the intent translation framework, providing semantic interpretation, policy grounding, and output generation. The final output is expressed in a structured format such as YAML and adheres to policy modeling defined in [TS-28.312].

Architecturally, the Intent Translator operates as a sequential pipeline composed of six logically distinct components. Each component handles a specific transformation step, beginning with user input and ending with policy document generation. The pipeline enables soft semantic matching, confidence scoring, and graceful handling of degraded translations. It is designed to support transparent, extensible, and standards-aligned translation of human goals into actionable configurations.

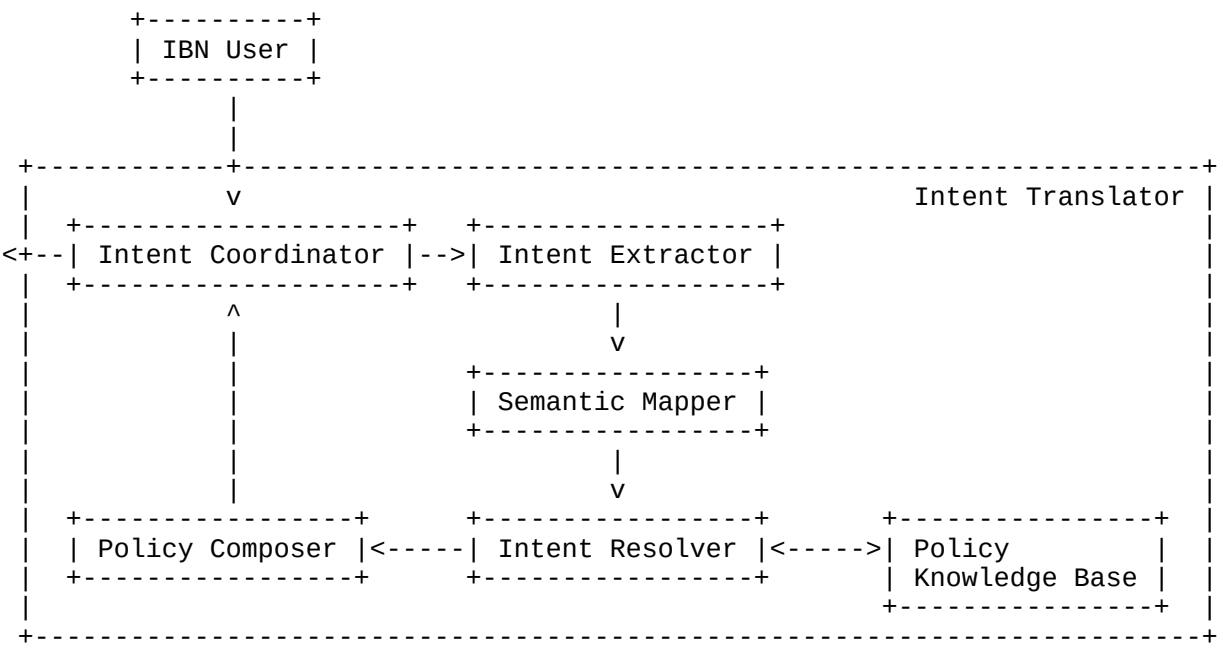

Figure 1: Intent Translator Component Architecture

The six main components of the Intent Translator are illustrated in Figure 1.

Intent Coordinator: An initial component of Intent Translator that receives user intent; dispatches and deploys. This component receives a natural language intent submitted by an IBN user or administrative interface. It forwards the natural language intent to the Intent Extractor. On the other hand, once the Policy Composer generates the final structured policy, the Intent Coordinator is also responsible for delivering it to downstream systems for enforcement, such as network controllers or orchestration engines.

Intent Extractor: A Few shot based large language model informed component that extracts structured elements from natural language. This component parses the incoming natural language statement to extract key semantic elements such as action, expactation object, and expactation target. These elements form the core of the structured intent representation and are passed to the Semantic mapper for KG embedding and semantic alignment.

Semantic Mapper: A component that projects structured intent into a semantic space aligned with domain knowledge. This component maps the structured intent fields into dense vector representations using a pre-trained embedding space. The embedding model is aligned with a domain-specific knowledge graph and allows the intent representation to be projected into the same semantic space used for stored policy facts. The aggregated intent vector is delivered to the reasoning module, Intent Resolver.

Intent Resolver: A Reasoning module that atches intent to policy triples; flags degraded matches. This component receives the embedded intent vector and compares it against the embeddings of knowledge graph triples stored in the Policy Knowledge Base. If a direct match is not available, soft matching is performed using semantic similarity scoring (e.g., cosine distance). When the similarity score for the best match falls below a defined threshold, the match is flagged as degraded. This degraded status is propagated downstream, enabling conditional processing and transparency in policy output.

Policy Knowledge Base: A knowledge base component that stores domain knowledge to provides embeddings and semantic structure. The Knowledge Base maintains the structured knowledge graph used throughout the translation process. It contains entity-relation triples that define valid intent mappings and service configurations. During reasoning, the Intent Resolver retrieves and compares Knowledge Base entries based on their semantic proximity to the intent vector. The Policy Knowledge Base supports approximate matching during inference.

Policy Composer: Generates YAML-formatted policy documents for deployment. The final component of the translation pipeline is responsible for synthesizing a policy document that aligns with [TS-28.312]. It uses both the extracted intent structure and the selected knowledge graph entry to construct a template based YAML-formatted policy.

Together, these components enable the reliable and transparent transformation of user-defined goals into system-aligned, deployable policies. The architecture is modular and extensible, allowing domain-specific enhancements without modification to the full translation pipeline.

3.2. Semantic Mapper

The Semantic Mapper translates structured intent fields into a unified semantic representation within an embedding space aligned with the Policy Knowledge Base. It serves as a semantic abstraction layer that enables approximate intent matching through latent space projection.

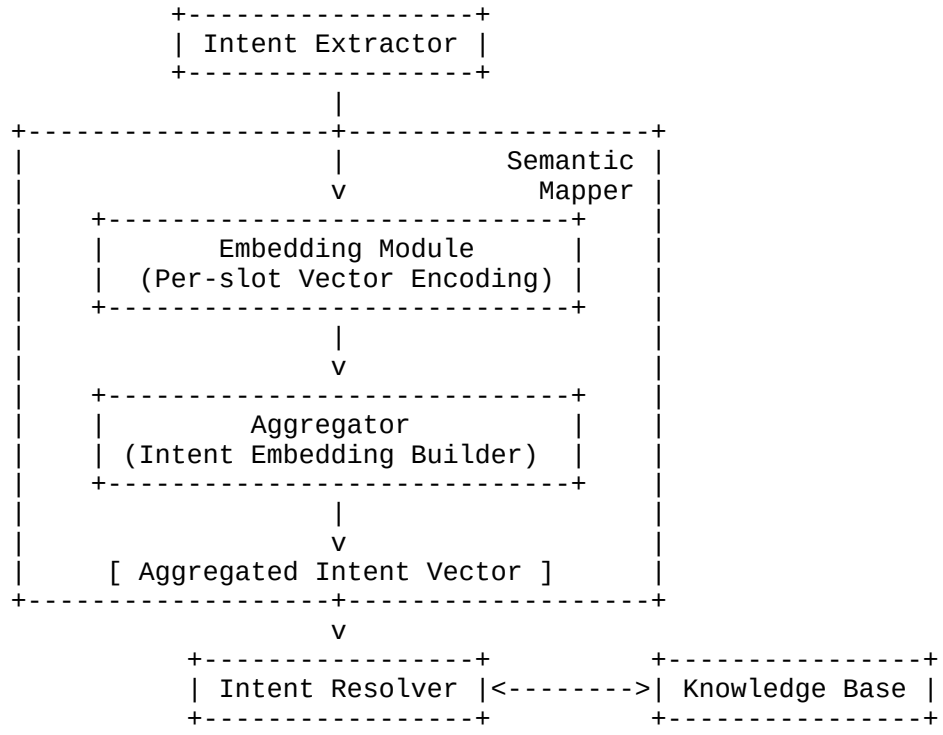

Figure 2: Semantic Mapper Internal Workflow

Figure 2 illustrates the internal flow of the Semantic Mapper. It begins with a structured intent delivered from the Intent Parser. The input undergoes semantic enrichment using a few-shot language model, followed by vector encoding of each semantic slot. The final stage aggregates the slot-wise vectors into a single intent embedding vector, which is then forwarded to the Intent Resolver.

The internal components of the Semantic Mapper are illustrated in Figure 2.

Embedding Module: A component that supports Per-slot Vector Encoding. Each enriched intent slots (e.g., action, expaction object, expectation target) is independently encoded as a dense vector in a shared semantic space. This allows for flexible alignment across synonymous or paraphrased expressions.

Aggregator: A component which builds intent embedding. This component aggregates the individual slot vectors into a single composite intent embedding. This intent embedding vector serves as a holistic semantic representation of the user’s goal and facilitates efficient semantic comparison during the reasoning phase.

The final output of the Semantic Mapper is the aggregated intent vector, which is passed to the Intent Resolver for policy resolution. This design supports soft matching and generalization over variations in language, domain vocabulary, and abstraction level.

3.3. Intent Resolver

The Intent Resolver is responsible for mapping the semantic representation of a user’s intent to a corresponding policy concept within the Policy Knowledge Base. It performs soft matching, evaluates confidence, and applies a thresholding mechanism to determine whether to generate a policy or flag the result for operator intervention.

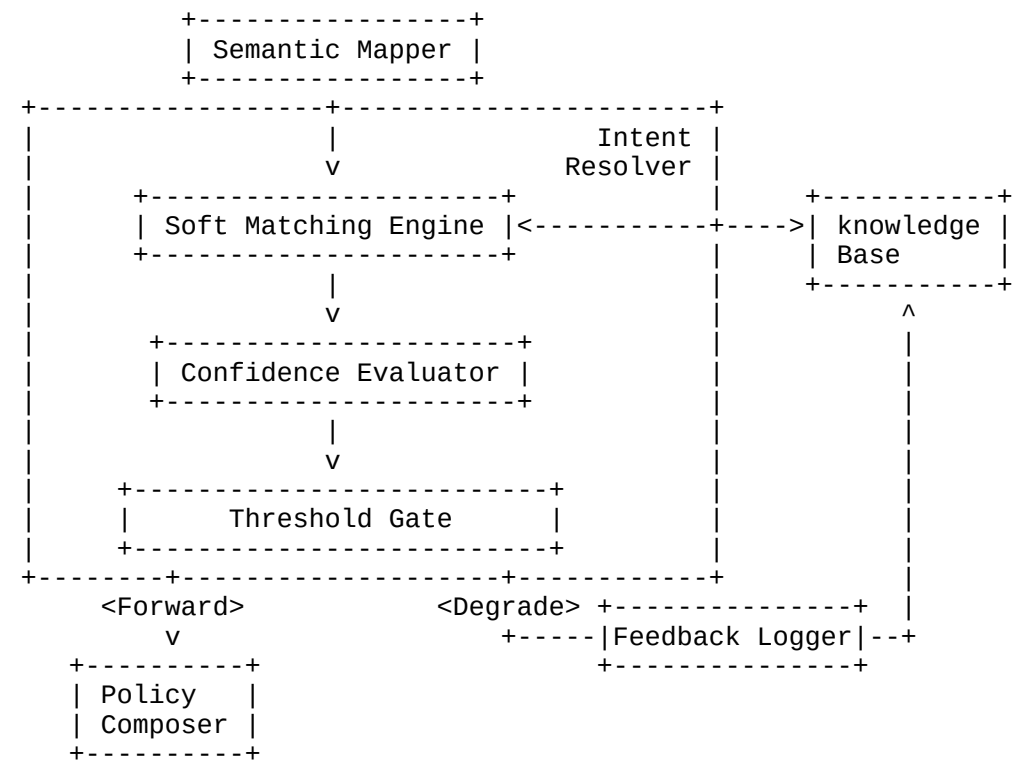

Figure 3: Intent Resolver Internal Architecture

The internal architecture of the Intent Resolver is shown in Figure 3.

Soft Matching Engine: This component receives the aggregated intent vector from the Semantic Mapper and computes its semantic similarity against all relevant policy entries stored in the Policy Knowledge Base. The comparison uses a vector-space similarity metric (e.g., cosine similarity) to identify the best-matching policy triple.

Confidence Evaluator: Once the most relevant policy candidate is selected, this component evaluates the semantic confidence score associated with the match. This score quantifies the degree of alignment between the user’s intent and the closest available domain policy.

Threshold Gate: The confidence score is evaluated against a configurable semantic threshold. If the score exceeds the threshold, the candidate policy is considered a valid match and is forwarded to the Policy Composer. If the score falls below the threshold, the intent is flagged as degraded.

Feedback Logger: For degraded matches, this component logs the failure for later analysis and may optionally initiate human-in-the-loop review. If confirmed or corrected, the outcome can be used to update the Policy Knowledge Base, thus improving future resolution accuracy.

The Intent Resolver enables explainable, flexible, and confidence-aware translation of semantic user goals into domain-aligned policy artifacts. It ensures that all generated outputs are grounded in interpretable knowledge while also supporting fallback and learning-based feedback.

The intent and high-level policy artifacts produced by the Intent Translator Framework can be expressed in standardized data models such as XML [RFC6020][RFC7950] or YAML [YAML]. These documents can be delivered to the appropriate management or orchestration systems via NETCONF [RFC6241], RESTCONF [RFC8040], or REST API [REST] interfaces for deployment and enforcement.

As described in the modular architecture of the framework, user-defined natural language intent is processed through a structured pipeline that includes the Intent Coordinator, Intent Parser, Semantic Mapper, Intent Resolver, and Policy Composer. The semantic reasoning within this pipeline is grounded in a domain-specific Policy Knowledge Base, enabling soft matching and degraded intent handling. This design ensures that operational goals - even when imprecisely expressed - can be semantically aligned to existing policy capabilities, enhancing automation readiness and interoperability across 6G-based IoT environments.

Therefore, this document proposes a practical and extensible architecture for intent translation in next-generation management systems. Through this architecture, high-level user goals can be reliably mapped to structured policy outputs, and network services can be automatically configured, validated, and adapted. The framework enables intent-based management to support scalable, knowledge-grounded automation in complex service domains such as IoT, vertical-specific networks, and intelligent edge infrastructures.

4. IANA Considerations

This document does not require any IANA actions.

5. Deployment Considerations

The deployment of the Intent Translator Framework requires alignment between the domain-specific Policy Knowledge Base and the operational policies supported by the underlying infrastructure. Domain-specific vocabularies, service models, and operational goals must be encoded within the knowledge base to ensure accurate semantic reasoning.

Operators should pre-train or validate the semantic embedding space against realistic intents and policy sets before enabling full automation. This is particularly important for service domains where intent ambiguity or synonymy could lead to unintended configurations.

6. Extensibility Considerations

The modular architecture of the framework allows for individual components - such as the Semantic Mapper or Policy Composer - to be adapted to different domains, languages, or knowledge graph models. As such, implementers can substitute the language model used for prompting, modify the embedding strategy, or replace the output schema (e.g., YAML, XML) without altering the end-to-end translation flow.

Additionally, the Policy Knowledge Base may be extended over time with new policy triples and relations to support evolving service capabilities. Such extensions should preserve backward compatibility by maintaining stable identifiers for core operational concepts.

7. Degradation and Human Oversight Considerations

The framework supports degraded intent resolution via soft matching and confidence scoring. While this enables flexible operation in the presence of vocabulary incompleteness or paraphrasing, operators should evaluate how degraded matches are handled within their intent lifecycle.

For high-assurance environments, degraded outputs should be reviewed by a human operator or routed to a validation pipeline before policy deployment. Logging mechanisms should be used to record degraded cases and their resolution outcomes to improve future model performance and policy reliability.

8. Security Considerations

The Intent Translation Framework must operate over authenticated and confidential channels (e.g., TLS/HTTPS) to prevent eavesdropping, message tampering, or replay attacks by malicious actors. Implementations should enforce strict certificate validation and regularly rotate cryptographic keys to maintain transport-layer security.

Because the system processes free-form natural language intents, it is vulnerable to adversarially crafted inputs designed to produce unintended or harmful policies. Deployments should incorporate input validation and semantic sanity checks such as confirmation interfaces for high-impact operations and rate limit or quarantine suspicious requests for manual review.

Intent collisions where contradictory or overlapping intents are submitted concurrently can lead to policy conflicts or enforcement gaps. The framework must include conflict-detection logic in its Policy Composer component and either automatically resolve detected collisions using a documented precedence model or escalate them to human operators for arbitration.

9. References

9.1. Normative References

[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/info/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/info/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/info/rfc7950>.

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

[RFC8329] Lopez, D., Lopez, E., Dunbar, L., Strassner, J., and R. Kumar, "Framework for Interface to Network Security Functions", RFC 8329, DOI 10.17487/RFC8329, February 2018, <https://www.rfc-editor.org/info/rfc8329>.

[RFC9315] Clemm, A., Ciavaglia, L., Granville, L. Z., and J. Tantsura, "Intent-Based Networking - Concepts and Definitions", RFC 9315, DOI 10.17487/RFC9315, October 2022, <https://www.rfc-editor.org/info/rfc9315>.

[RFC9365] Jeong, J., Ed., "IPv6 Wireless Access in Vehicular Environments (IPWAVE): Problem Statement and Use Cases", RFC 9365, DOI 10.17487/RFC9365, March 2023, <https://www.rfc-editor.org/info/rfc9365>.

9.2. Informative References

[I-D.ietf-i2nsf-applicability]
Jeong, J. P., Hyun, S., Ahn, T., Hares, S., and D. Lopez, "Applicability of Interfaces to Network Security Functions to Network-Based Security Services", Work in Progress, Internet-Draft, draft-ietf-i2nsf-applicability-19, 3 April 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-i2nsf-applicability-19>.

[I-D.jeong-i2nsf-security-management-automation]
Jeong, J. P., Lingga, P., Jung-Soo, J., Lopez, D., and S. Hares, "An I2NSF Framework for Security Management Automation in Cloud-Based Security Systems", Work in Progress, Internet-Draft, draft-jeong-i2nsf-security-management-automation-08, 26 July 2024, <https://datatracker.ietf.org/doc/html/draft-jeong-i2nsf-security-management-automation-08>.

[I-D.jeong-nmrg-ibn-network-management-automation]
Jeong, J. P., Ahn, Y., Gu, M., Kim, Y., and J. Jung-Soo, "Intent-Based Network Management Automation in 5G Networks", Work in Progress, Internet-Draft, draft-jeong-nmrg-ibn-network-management-automation-06, 9 June 2025, <https://datatracker.ietf.org/doc/html/draft-jeong-nmrg-ibn-network-management-automation-06>.

[I-D.yang-i2nsf-security-policy-translation]
Jeong, J. P., Lingga, P., and J. Yang, "Guidelines for Security Policy Translation in Interface to Network Security Functions", Work in Progress, Internet-Draft, draft-yang-i2nsf-security-policy-translation-16, 7 February 2024, <https://datatracker.ietf.org/doc/html/draft-yang-i2nsf-security-policy-translation-16>.

[YAML] Ingerson, B., Evans, C., and O. Ben-Kiki, "Yet Another Markup Language (YAML) 1.0", Available: https://yaml.org/spec/history/2001-05-26.html, October 2023.

[TS-23.501]
"System Architecture for the 5G System (5GS)", Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3144, September 2023.

[TS-28.312]
"Intent Driven Management Services for Mobile Networks", Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3554, September 2023.

[TR-28.812]
"Study on Scenarios for Intent Driven Management Services for Mobile Networks", Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3553, December 2020.

[TS-23.288]
"Architecture Enhancements for 5G System (5GS) to Support Network Data Analytics Services", Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3579, September 2023.

[TS-29.520]
"Network Data Analytics Services", Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3355, September 2023.

[RFC7149] Boucadair, M. and C. Jacquenet, "Software-Defined Networking: A Perspective from within a Service Provider Environment", RFC 7149, March 2014, <https://www.rfc-editor.org/rfc/rfc7149>.

[USENIX-ATC-Lumi] Jacobs, A., Pfitscher, R., Ribeiro, R., Ferreira, R., Granville, L., Willinger, W., and S. Rao, "Hey, Lumi! Using Natural Language for Intent-Based Network Management", USENIX Annual Technical Conference, Available: https://www.usenix.org/conference/atc21/presentation/jacobs, July 2021.

[BERT] Devlin, J., Chang, M., Lee, K., and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding", NAACL-HLT Conference, Available: https://aclanthology.org/N19-1423.pdf, June 2019.

[REST] Author, A., "REST API for Network Management", Available: https://example.com/rest-spec, July 2025.

[Deep-Learning] Goodfellow, I., Bengio, Y., and A. Courville, "Deep Learning", Publisher: The MIT Press, Available: https://www.deeplearningbook.org/, November 2016.

[Survey-IBN-CST-2023] Leivadeas, A. and M. Falkner, "A Survey on Intent-Based Networking", Available: https://ieeexplore.ieee.org/document/9925251, March 2023.

Appendix A. Acknowledgments

This work was supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea Ministry of Science and ICT (MSIT) (No. RS-2024-00398199 and RS-2022-II221015).

This work was supported in part by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea Ministry of Science and ICT (MSIT) (No. 2025-RS-2022-II221199, Regional strategic industry convergence security core talent training business).

Changes from draft-gu-nmrg-intent-translator-00 The following changes are made from draft-gu-nmrg-intent-translator-00: * This version is submitted for the maintanance of draft-gu-nmrg-intent-translator.

Authors' Addresses

Mose Gu (editor)
Department of Computer Science and Engineering
2066 Seobu-Ro, Jangan-Gu
Suwon
Gyeonggi-Do
16419
Republic of Korea
Email: rna0415@skku.edu

Jaehoon Paul Jeong (editor)
Department of Computer Science and Engineering
2066 Seobu-Ro, Jangan-Gu
Suwon
Gyeonggi-Do
16419
Republic of Korea
Phone: +82 31 299 4957
Email: pauljeong@skku.edu

Yoseop Ahn
Department of Computer Science and Engineering
2066 Seobu-Ro, Jangan-Gu
Suwon
Gyeonggi-Do
16419
Republic of Korea
Phone: +82 31 299 4106
Email: ahnjs124@skku.edu
URI: http://iotlab.skku.edu/people-Ahn-Yoseop.php