ETSI GS ENI 033 V4.1.1 (2024-08)

Experiential Networked Intelligence (ENI); Definition, Requirements and Procedure of Intent Policy Multi-Stage Translating

Disclaimer

Reference
DGS/ENI-0033v411_MSTranslating

Keywords
intent management, policy management

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - APE 7112B
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° w061004871

Contents

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The declarations pertaining to these essential IPRs, if any, are publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (https://ipr.etsi.org/).

Pursuant to the ETSI Directives including the ETSI IPR Policy, no investigation regarding the essentiality of IPRs, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

DECT™, **PLUGTESTS™**, **UMTS™** and the ETSI logo are trademarks of ETSI registered for the benefit of its Members. **3GPP™** and **LTE™** are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners. **oneM2M™** logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners. **GSM**® and the GSM logo are trademarks registered and owned by the GSM Association.

Foreword

This Group Specification (GS) has been produced by ETSI Industry Specification Group (ISG) Experiential Networked Intelligence (ENI).

Modal verbs terminology

In the present document "**shall**", "**shall not**", "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

Executive summary

The present document specifies a high-level functional abstraction of the process of intent policy Multi-Stage translating in ENI system in terms of Functional Modules, Internal Reference Points and working pipelines.

Introduction

The present document defines a high-level functional abstraction of Intent Policy Multi-Stage Processing. The organization of the present document is as follows. Clause 1 defines the scope of the present document. Clauses 2 and 3 provide normative and informative references and definition of terms, respectively. Clause 4 provides an informative overview of Intent Policy Multi-Stage Translating, including its motivation, benefits, important concepts and an overview of its Functional Modules. Clause 5 defines important design principles of the processing. Clause 6 provides some use cases of Intent Policy Multi-Stage Processing. Clause 7 gives away some potential future works on the present document.

1 Scope

The present document augments existing intent policy translating procedure in ENI. The purpose of the present document is to describe intent policy multi-stage translating in ENI system, and to enhance intent policy multi-stage translating.

The present document also defines the output(s), input(s), internal process and interaction of every stage during intent policy multi-stage translating.

Intent policy multi-stage translating is a detailed procedure that can translate an intent policy according to the Policy Continuum. There is an external knowledge base to be added to provide a set of multi-stage general processing scheme for intent policy.

2 References

2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at https://docbox.etsi.org/Reference.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are necessary for the application of the present document.

[1] ETSI GS ENI 005 (V3.1.1): "Experiential Networked Intelligence (ENI); System Architecture".

[2] ETSI GS ENI 030 (V4.1.1): "Experiential Networked Intelligence (ENI); Transformer Architecture for Policy Translation".

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] ETSI GS ENI 019 (V3.1.1): "Experiential Networked Intelligence (ENI); Representing, Inferring, and Proving Knowledge in ENI".

3 Definition of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the following terms apply:

AI model (in the present document): model that is capable of processing the understanding and generation of natural language

business intent: abstract network intent input by the user

intent-level entity: specific noun element in an intent-level template that represents a network attribute

intent-level template: network intent with fixed format

knowledge base: unified repository encompassing diverse structural resources including knowledge graphs for linking data contexts and text documents

knowledge graph: data organization model leveraging graph theory and logical frameworks to depict the interconnectedness and logical associations within information, realized through a graphical structure for coherent knowledge storage and handling

named entity: word or phrase that refers to an item or process of interest

named entity recognition: information extraction task focused on identifying specific, named elements within text data

network entity: group of network information combination, such as [time and bandwidth], [start time, end time and packet loss], etc.

network policy: Domain Specific Language (DSL) generated from a user-level template

on-demand service: service that is provisioned and used as needed

part-of-speech tagging: natural language processing technique used to determine the grammatical category of each word in a sentence, such as nouns, verbs, adjectives, etc.

service-level template: template incorporating Quality of Service, Access Control List, Service-Level Agreements, and Network Function Virtualisation, etc.

user-level template: service-level template that incorporates user preferences, device information, etc.

3.2 Symbols

Void.

3.3 Abbreviations

For the purposes of the present document, the following abbreviations apply:

ACL	Access Control List
AI	Artificial Intelligence
CRF	Conditional Random Fields
DSL	Domain Specific Language
HMM	Hidden Markov Model
NER	Named Entity Recognition
NFV	Network Functions Virtualisation
NLP	Natural Language Processing
QoE	Quality of Experience
QoS	Quality of Service
RNN	Recurrent Neural Network
SLA	Service Level Agreement

VNF Virtual Network Function

4 Overview of Intent Policy Multi-Stage Processing

4.1 Introduction

This clause provides an informative introduction to Intent Policy Multi-Stage Processing in the ENI Policy Management Functional Block of ETSI GS ENI 005 [1]. Clause 4.2 describes the background and motivation of Intent Policy Multi-Stage Processing, and then provides a high-level description of Intent Policy Multi-Stage Processing in the ENI system. Clause 4.3 describes the functional architecture of Intent Policy Multi-Stage Processing in terms of Processing Stages. Clause 4.4 introduces each step of Intent Policy Multi-Stage Processing.

4.2 Description of Intent Policy Multi-Stage Processing

4.2.1 Background

With the development of the research of the sixth-generation wireless communication network (6G), the network business scenarios are increasingly diversified. This evolution is significantly propelled by the synergistic integration of artificial intelligence with communication technology, which serves as a pivotal pillar supporting the emergence of novel service paradigms and further driving the development of the network.

The key challenge of current 6G research is to achieve on-demand services in all scenarios. This "on demand" concept is fundamentally different from the traditional service provisioning model, where services are pre-established and are typically locked in for a fixed duration. On-demand service refers to the way of providing customized network services to meet different needs by efficiently allocating network resources across multiple domains according to the specific characteristics and needs of network scenarios and services. The core of its realization lies in transforming the traditional network architecture model and establishing a new service-centric model to enhance the comprehensiveness of network services, optimize the efficiency of resource utilization, and ensure that complex and changing personalized scenarios are accurately matched with services, thus maximizing the value of the network. In order to achieve this vision, the network is capable of adapting to the subtle and specific needs of users. This requires a complex understanding of user intent, including the ability to quickly extract prominent information and translate it into precise network strategies. This capability is the foundation of the vision for a fully automated network. Furthermore, the core of the broader objectives of 6G technology is to pursue unparalleled user experiences for all parties, including vertical industries and individual consumers, through the seamless integration of user intent comprehension and dynamic network adaptation.

4.2.2 High-Level Description of Intent Policy Multi-Stage Processing

To accommodate this transition to on-demand services, the intent policy multi-stage processing framework became a key component of the 6G ecosystem. This innovative mechanism aims to interpret and manipulate user intent and translate them into executable network policies that dynamically adapt to the fluctuating demands of different network scenarios. This framework employs a hierarchical decomposition methodology to deconstruct the user's natural language intent into multiple levels. This ensures the accurate, timely and standardized translation of a natural language intent policy into a multi-level fine-grained network template that supports on-demand services.

NOTE: This hierarchical decomposition methodology is similar to how policy grammar is structured in ETSI GS ENI 030 [2] and modelled in ETSI GS ENI 019 [i.1].

More specifically:

- **Multi-level Fine-grained Network Template:** A hierarchical network configuration template system, including intent-level templates, service-level templates, and user-level templates, enables level-by-level refinement and personalized configuration.

Each level of the intent policy multi-stage processing framework represents a deeper and more detailed representation of the user's intent and is customized to support on-demand services. It acts as a bridge that seamlessly connects the human-centric realm of expression with the intricate configuration of the network, ensuring that the network infrastructure dynamically adapts to meet the precise and changing needs of users.

4.3 Functional Architecture

Intent Policy Multi-Stage Processing transforms the network requirements in the user's natural language intent into identifiable and deliverable network policies. Figure 4.3.1 shows the transformation of intent policy, which consists of four steps: generate intent-level templates, generate service-level templates, generate user-level templates, and generate network policies.

Figure 4.3.1: Transformation of Intent Policy

The detailed definition of the templates and policies for the Intent Policy Multi-Stage Processing is as follows:

- **Intent-level Template:** An initial state of a network policy for mining user intent into intent-level entities and combining them into templates via the knowledge base (e.g. a document) and AI model. The format of intent-level template has certain restrictions, and it is generally in the form:<Operation> <Entity Type>: Target Entity, <Operation><Entity Type>: Target Entity, etc.

- **Operation:** Operation represents the action or activity that the user intends to perform on or with the Target Entity within the context of their expressed intent. It signifies the type of task, command, or manipulation the user desires, such as 'add', 'set', 'block', 'allow', etc.

- **Entity Type:** A classification of specific information or object types that represent the key elements extracted from user intent. All intent-level entity types include: Object, Middlebox, Location, Service, Time, Bandwidth, Src, Dst, Protocol.

- **Target Entity:** Target Entity refers to the specific object, concept, or piece of information that the user is directly referring to or interacting with in their expressed intent, serving as a key point for action or retrieval in a system. For example, 'students' under <Object>, (firewall,admit) under <Middlebox>, 'video streaming' under <Service>, and [10 Mbps, 20 Mbps] under <Bandwidth>.

NOTE 1: Not every <Entity Type> is preceded by an <Operation>, e.g. for Object, Time, etc. no <Operation> is required.

NOTE 2: Intent-level entities could be expressed in a variety of ways and are not limited to string type expressions, e.g. tuple form under the <Middlebox> type: (firewall,close), list form under the <Bandwidth> type: [10 Mbps, 20 Mbps].

- **Service-level Template:** An intermediate state network policy, which can select the appropriate service-level templates according to the extracted intent-level entity types and user intent through the AI model, and form an information structure that meets the business needs and user group characteristics according to the information of the knowledge base. This typically consists of Quality of Service (QoS) template, Access Control List (ACL) template, Service-Level Agreement (SLA) template, and Network Functions Virtualisation (NFV) template.

- **QoS Template:** An intermediate state network policy framework for defining and guaranteeing performance levels for specific traffic or services in a network. It typically includes presets for parameters such as bandwidth, latency, etc. to ensure that critical applications are prioritized on the network to meet user performance requirements.

- **ACL Template:** An intermediate state network security policy framework for controlling access to network traffic. It defines which source IP addresses access which destination IP addresses and at what times. ACL templates also include rules for handling intermediate network devices, such as firewalls, to allow or deny specific types of traffic to protect the network from unauthorized access and other security threats.

- **SLA Template:** A standardized framework of agreements used to clarify expectations between service providers and customers regarding service quality and availability. It typically includes service level objectives and service credit or compensation provisions in the event that these objectives are not met.

- **NFV Template:** A technical framework for defining how to deploy and manage network functions on a standardized virtualized infrastructure. It includes guidance on the configuration and management of virtualized network devices and how to implement traditional hardware network functions in a cloud environment.

- **User-level Template:** One that combines knowledge base (e.g. knowledge graph) and differentially adapts decisions based on personalized information such as user devices, ultimately generating customized intermediate network policies. User-level intent templates contain multiple user sub-templates, each of which corresponds to a specific user and together form a multi-user QoE model.

NOTE 3: Differentiation adjustment mainly includes the underlying network operational objects and the fine-grained quantified indicators or network operations expected by the underlying network corresponding to the previous layer template.

NOTE 4: The above types may be modelled in a future release of ETSI GS ENI 019 [i.1].

- **Network Policy:** Network Policy is a deployable Domain Specific Language (DSL) [1] that enables administrators to define and enforce rules governing network behaviour within a system or infrastructure.

Translating user intent into executable network policies is a complex task that requires not only a grasp of the user's explicit request, but also a deep understanding of the underlying context and semantics. Named Entity Recognition (NER) and Knowledge Graphs play an important role in this regard [2]. The application of them is an essential part of each of the above steps. Named entities are objects that can be identified with a proper name, and generally refer to elements with specific meanings. For example, this includes names of people, places, names of organizations, proper nouns and so on. The purpose of NER is to extract the above elements from unstructured input text, and to identify more types of elements according to business requirements, such as product name, model, price, etc. A knowledge graph constitutes a graphical depiction technique linking interconnected information via nodes and edges. It encompasses an extensive collection of items and their associations. This methodology enables the clear illustration of intricate connections among these items, facilitating a deeper comprehension of the data and revealing underlying patterns and regularities within it.

The overall steps in the framework of the Intent Policy Multi-Stage Processing are as follows:

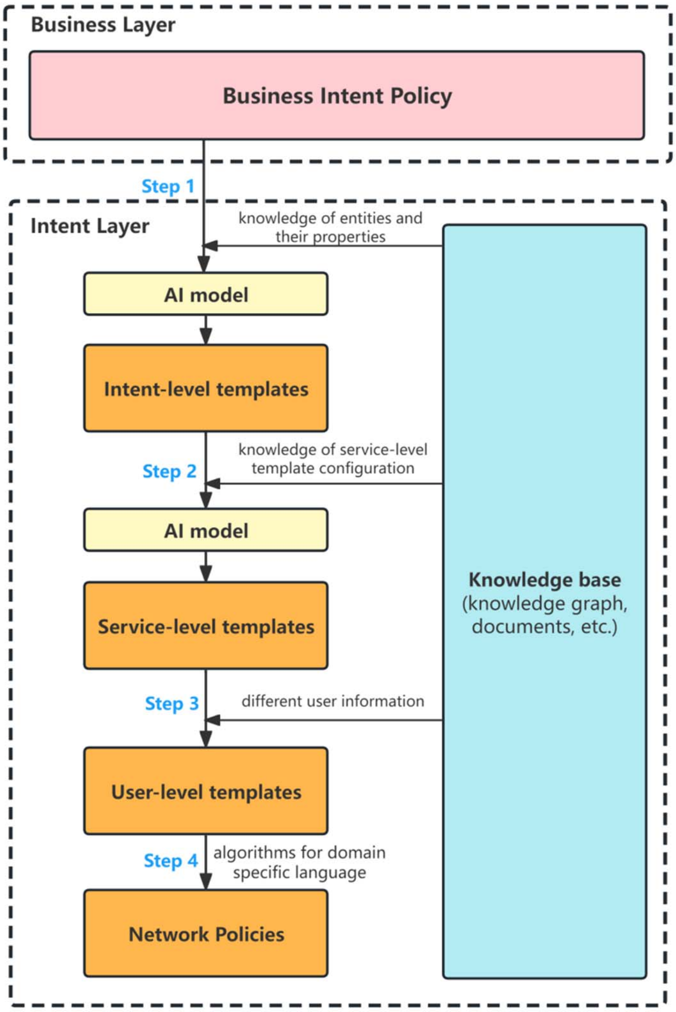
- Step 1: Intent-level entities along with their attributes are primarily derived from the AI model. The AI model generates accurate and contextually relevant intent-level template by incorporating the knowledge base.

- Step 2: The AI model selects an appropriate service-level template based on the intent-level entity types as well as the information in the intent-level template. The knowledge base is used as a resource to populate specific details in the service-level templates.

- Step 3: Service-level templates are populated with device information from the knowledge base, and then converted to user-level templates to achieve optimal QoS.

- Step 4: Network policies are generated based on user-level templates.

Figure 4.3.2 shows a more detailed flow from user intent to network policies during Intent Policy Multi-Stage Processing.



NOTE: Either two different AI models or a single AI model could be used, depending on the application and use case.

Figure 4.3.2: Intent Policy Multi-Stage Processing Architecture Diagram

The detailed steps in the framework of the Intent Policy Multi-Stage Processing are as follows:

- Step 1: The natural language intent entered by the user combined with the latest information from the knowledge base is fed into the AI model to obtain the corresponding intent-level entity in the network and intent-level templates in the network.

- Step 2: The AI model selects the appropriate service-level template based on the intent-level entity types as well as the information in the intent-level template and populates some of the configuration information of the service level template based on the information in the knowledge base.

NOTE 4: When the Knowledge Base populates a service-level template, it does not populate all of the information, only the configuration information required for that layer of the template. For example, the access control list template of the online meeting service helps users connect to the primary server, and the service-level agreement template allocates resources such as bandwidth to users.

- Step 3: User-level templates are adapted by querying the knowledge base for user devices and user preferences. The queried information is populated directly into the different user-level templates using a populating algorithm.

- Step 4: Convert user-level templates into network policies using specific algorithms.

4.4 Procedures of Intent Policy Multi-Stage Processing

4.4.1 Generate Intent-Level templates

4.4.1.1 Overview

This clause describes how an AI model combined with a knowledge base transforms abstract user intent into the corresponding intent-level template. The benefits of this approach and the requirements are also described.

Because users typically state complex online needs and some subtle expressions of demand in a single input process, so the AI model trained on a specific network domain corpus are essential, but since the continuous development of the network may lead to outdated information in the models, the impact of outdated information in the models is effectively reduced by introducing a knowledge base.

In the process of the Intent Policy Multi-Stage Processing, the main task of generating intent-level templates is to identify the key vocabulary and semantic information based on the user input text information, and then combine the above information into a complete intent-level template.

Specifically, at this stage, the text input by the user is queried by the knowledge base and the latest information is returned, and the user's natural language intent is combined to form a new prompt input to the AI model, generating a complete intent-level template to better understand the user's intent. Intent-level templates contain a variety of keywords, such as operation, target entity, service, etc.

The purpose of generating intent-level templates is to be able to more accurately understand the user's abstract intention, and to provide the basis for subsequent processing, which directly affects subsequent processing and the user experience. A good intent template should have the following characteristics:

- Be able to clearly express the user's intent;
- Contains enough information to guide subsequent processing;
- Has a certain flexibility, and can adapt to different scenarios and user needs.

4.4.1.2 The Sequence Generation Method

Sequence generation is a Natural Language Processing (NLP) task [2] that aims to generate a sequence or text based on given conditions or contextual information. Sequence generation is used in many different application areas, including dialogue systems, summary generation, machine translation, text classification, etc.

Sequence-generated tasks are divided into two types:

- Conditional sequence generation is the generation of a sequence or text based on a given condition. These conditions could be input text, images, speech, etc., as well as generated output sequences or text. Conditional sequence generation is often used in dialog systems, question answering systems, text classification and other tasks.
- Unconditional sequence generation is the generation of sequences or text without regard to any conditions or context information. This type of task is often used in applications such as text summarization.

In sequence generation, different methods and techniques are used to generate sequences or text, including rule-based methods, probability-based methods, and neural network-based methods.

A rule-based approach uses predefined rules to generate sequences or text. These rules can be formulated according to domain knowledge and linguistic knowledge, such as grammar rules, vocabulary collocation rules, semantic rules, etc. Rule-based approaches are often appropriate for specific tasks and domains, and require specialized knowledge and experience to develop rules.

Probability-based methods use probabilistic models to generate sequences or text. These models typically use Hidden Markov Models (HMMs), Conditional Random Fields (CRFs), or probability graph models to model the probability distribution of a sequence. By calculating probabilities, the best generated sequence or text can be found and optimized and adjusted. Probability-based methods are generally suitable for tasks with short sequences, such as named entity recognition, part-of-speech tagging, etc.

Neural network-based approaches use deep learning techniques to generate sequences or text. These techniques typically use Recurrent Neural Networks (RNNs) or Transformer models to learn representations of input sequences or text and use those representations to generate output sequences or text. Neural network-based methods are generally suitable for tasks with long sequences, such as machine translation, text classification, etc.

4.4.1.3 Operation Process

- **Input:** Natural language intent related to network requirements (i.e. business intent). For example, The online meeting of all students will be held at 9:00 today.

NOTE: The user's natural language intent consists of unstructured text, which the process will convert into a structured form that a network device can execute.

- **Process:** The text entered by the user combined with the latest information from the knowledge base is processed by an AI model to better understand the user's intent. The process includes entity recognition, attribute extraction and template combination. The goal of entity recognition and attribute extraction is to accurately identify and extract key intent-level entities that are closely related to the user's intent, which includes elements such as Operation (e.g. add, set), Object (e.g. student, teacher), Location (e.g. school, office building), and Service (e.g. online meeting, file transfer). The role of template assembly is to organically combine these intent-level entities and their attributes in a fixed format to form a data template that is both structured and contains complete information.

- **Output:** Intent-level templates generated from user intent, for example: <Object>: all students, <Allow><Service>: online meeting, <Time>: 9:00.

4.4.2 Generation of Service-Level Templates

4.4.2.1 Overview

This clause describes how the AI model selects the corresponding service-level templates based on the intent-level entity types and attributes in the intent-level template and generates service-level templates with specific configurations by querying the knowledge base.

In complex network scenarios, routing control, resource allocation, and VNF deployment coordination are often required to achieve network tasks. Different network architecture levels and adjustment requirements require different policy descriptions. The AI model predicts and selects the appropriate template based on the intent-level template generated in the previous step. In the present document, a service-level template can be divided into various templates such as Quality of Service (QoS) template, Access Control List (ACL) template, Service-Level Agreement (SLA) template and Network Functions Virtualisation (NFV) template. The following takes the QoS template and ACL template as examples to illustrate the process of generating a service level template.

4.4.2.2 Operation Process

- **Input:** The intent-level template generated in the previous step. For example, <Object>: all students,<Allow><Service>: online meeting, <Time>: 9:00.

- **Process:** First determine the service requirement and service-level template type according to the intent-level template, for example, the service-level template types are QoS template and ACL template according to the input example. Subsequently, the corresponding service-level template are adhered to, in accordance with the intent-level template. Then, based on the intent-level template, the knowledge base is queried to obtain the necessary information about different service-level templates, such as bandwidth information, enabling and disabling the firewall, receiving or discarding packets, etc. Finally, different service-level templates are grouped together.

- **Output:** A patchwork of different service-level templates. For example, QoS template: <Object>: all students, <Service>: online meeting, <Time>: 9:00, <Bandwidth>: [20 Mbps, 100 Mbps]; ACL template: <Src>: IP of all students, <Dst>: IP of the server, <Time>: 9:00, <Middlebox>:(firewall, admit).

4.4.3 Generation of User-Level Templates

4.4.3.1 Overview

This clause describes how to generate user-level templates based on service-level templates.

Intent-level entities extraction combined with general service standard method can determine the current service-level templates, but because in the same service, different user equipment, user preferences, etc., can also bring the different user demands and experience. This makes it difficult to handle the differences between different user instantiations. Through the introduction of knowledge base, more refined information (such as IP address, etc.) is provided for different users, which can be highly customized to better meet the specific needs of user groups.

4.4.3.2 Operation Process

- **Input:** The service-level template generated in the previous step and the results returned by the knowledge base. For example, service-level template: QoS template: <Object>: all students, <Service>: online meeting, <Time>: 9:00, <Bandwidth>:[20 Mbps, 100 Mbps];the results returned by the knowledge graph: <Student 1>: IP index of student 1,<Student 2>: IP index of student 2, etc.

- **Process:** There are four main steps to creating user-level templates:

  - Step 1: Introduce the external knowledge graph to assist in building a multi-user QoE model.

  - Step 2: Decompose the service-level templates at the user layer, and adjust the decision differently according to the IP addresses of different users in the previous layer template.

  - Step 3: For the necessary information in the underlying network transmission layer, adjust the index of the user's IP address and the Bandwidth parameters according to the user device, network status and user preference (see figure 4.4.3.2).

NOTE: Each user's IP address is mapped to the respective device, network status, and user preferences for network transport layer configuration.

  - Step 4: Generate user-level templates based on external information.

- **Output**: Generate user-level templates that are appropriate for different users and devices. For example, user template 1:<Object>: IP index of student 1, <Service>: online meeting, <Time>: 9:00, <Bandwidth>:[25 Mbps, 100 Mbps]; user template 2:<Object>: IP index of student 2, <Service>: online meeting, <Time>: 9:00, <Bandwidth>:[30 Mbps, 120 Mbps]; user template 3:<Object>: IP index of student 3, <Service>: online meeting, <Time>: 9:00, <Bandwidth>:[15 Mbps, 80 Mbps], etc.

Here is an example of this clause:

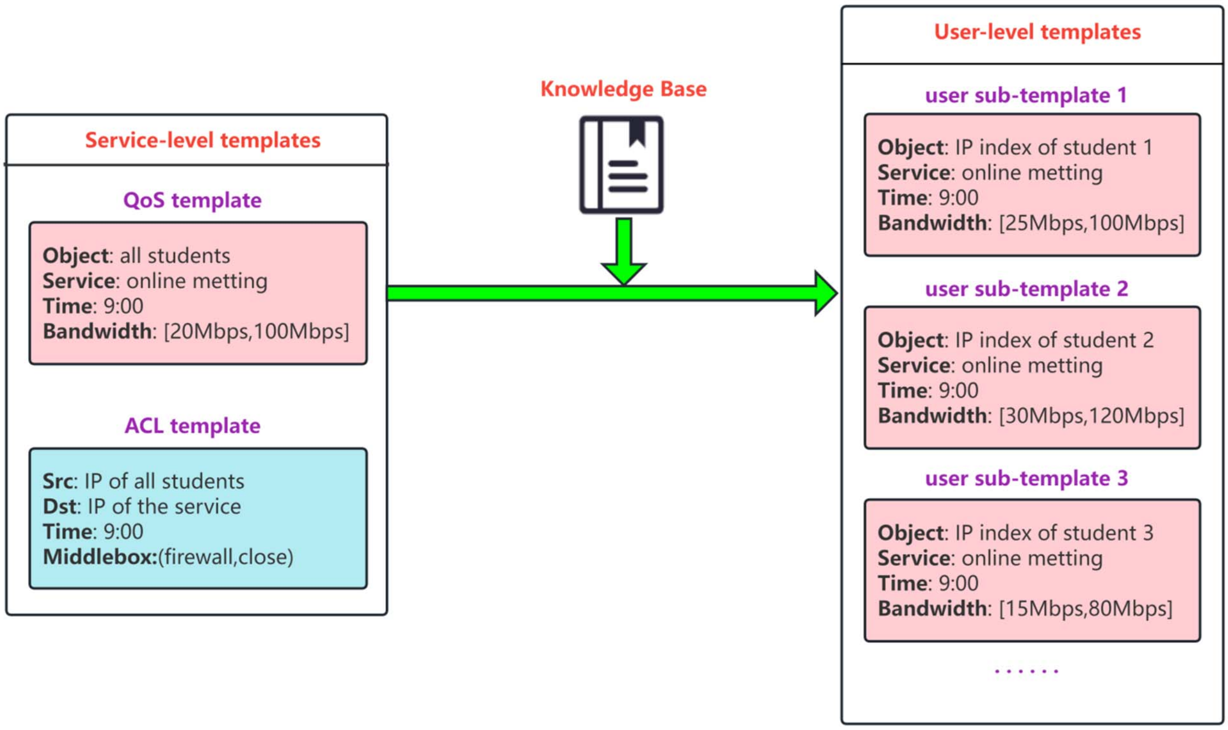


Figure 4.4.3.2: Example of generating user-level templates

4.4.3.3 Benefit

The introduction of user-level templates during intent policy multi-level processing has several benefits:

1) It enables the strategy generation module to make differentiated decisions at the user level and ensure highly customized user-level templates to meet the specific needs of different user groups.

2) It reduces the time and computational cost of customized configurations for each user, provides high-quality services, and maximizes the efficiency of network resources.

3) Using external knowledge, the framework is portable and scalable to quickly adapt to a variety of personalized network service scenarios.

4.4.4 Generation of Network Polices

In the intent policy multi-level processing stage, the final step is to generate the corresponding Domain Specific Language (DSL) according to the user-level templates generated in the previous step, and the details will be expanded in the following clauses.

5 Intent Policy Multi-Stage Processing Design Requirements

5.1 Introduction

The following clauses define functional and non-functional requirements of Intent Policy Multi-Stage Processing, respectively.

NOTE: In Release 4, the clauses focus on the knowledge base, the form of the intent policy.

5.2 Functional Requirements for Intent Policy Multi-Stage Processing

1) The ability to deal with complex or abstract intent policies. Intent policy multi-stage processing is able to identify and extract relevant information from different types of user intents.

2) The ability to extract translate intents into intent-level entities and establish intent templates. The system is able to use an AI model to mine the network elements and output them into a series of key intent-level entities labelled with the corresponding intent-level entity category to establish the corresponding intent templates.

3) The ability to predict and select the corresponding service-level template according to the demands. The system is able to predict different policy description criteria based on the output intent-level entity category, intent-level entity content, and different network architectures and adjustment requirements, and then choose a more accurate service-level template.

4) The ability to enhance the output of AI models based on external knowledge bases, including knowledge graphs. The system uses this technique in all three stages of the generation process, where the external knowledge base enables the AI model to generate more accurate intent-level templates, service-level templates, and user-level templates by returning accurate and reliable information and updating the prompt.

5) The ability to create a multi-user QoE or QoS model according to the current different user devices and user preferences, and make differentiated adjustments to the template of the previous layer.

6) The ability to generate corresponding network policies according to the service-level templates. The system is able to identify the corresponding network task based on the output user-level template and translate it into the corresponding network configuration and policies.

5.3 Non-Functional Requirements for Intent Policy Multi-Stage Processing

- **Reliability:** The system ensures high availability and stability to prevent user data loss caused by faults.
- **Scalability:** The system is able to support large-scale users and data, and can be flexibly expanded with the development of the business.
- **Security:** The system ensures the security and privacy of data to avoid the disclosure of sensitive information and attacks.
- **Maintainability:** The system is easy to maintain and upgrade, including code readability, reusability, testability and other aspects.
- **Ease of use:** The system provides a friendly user interface and interaction methods, so that users can easily use the system.
- **Performance:** The system has high performance and efficiency, and can quickly process a large number of requests and data.

6 Use Cases

6.1 Use Case: User intent to the network polices

6.1.1 Use case context

This use case covers the entire process from a user creating an intent policy to the generation of corresponding network policies. From the initial abstract intent input by the user, to the intermediate network service templates, which are divided into service-level and user-level templates, to the final network policies, each step refines the previous step, while also laying the foundation for the generation of the next step and providing clearer guidance.

6.1.2 Description of the use case

6.1.2.1 Overview

The user inputs natural language intent into the system so that the system can generate network policies that can be issued based on the user's needs, preferences, and current network environment. The process includes:

- Inputting user intent. Compared to the specific configuration and parameter information entered by the network administrator in traditional cases, user intent only contains an abstract requirement expressed in natural language. For example, "I want to watch live football matches at 7 p.m.".
- Generation of intent-level template. The parameters of intent-level template have similarities with the specific parameters and configurations mentioned in the previous step as input by the network administrator, but they cannot be applied directly. They can be considered as a representation of abstract user requirements, forming intermediate template descriptions with certain formats and expressive capabilities to adapt to different types and formats of user input.
- Generation of service-level template. Based on the intent level template determined in the previous step, the template is further transformed by considering the types of intent-level entities in it and their attributes. It has more information about network environments and business scenarios and is mainly divided into QoS templates and ACL templates.
- Generation of user-level template. This template is based on the service-level template and takes into account user diversity, device diversity, and network environment diversity, formulating templates that best meet the needs of each user.
- Generation of network policies. This step extracts and combines information from the service-level templates through algorithms to form a Domain Special Language (DSL), which is the network policy.

6.1.2.2 Motivation

Under the traditional network management model, enterprises need to hire professional network administrators to configure and manage the network. These administrators need to have deep network knowledge and experience, and be able to skilfully input various parameter information to ensure the normal operation of the network. However, this approach not only puts high demands on the human resources of the enterprise, but also often makes the management process cumbersome and complex.

However, with the support of AI models, network management promises to be more intelligent and efficient, potentially reducing reliance on skilled professionals. The key to the implementation of this intelligent network management method lies in the way AI models handle input information. In traditional network management, administrators need to input specific parameter information for configuration and management. More importantly, they need associated knowledge (e.g. from network telemetry) to understand if the service is operating correctly. The approach in the present document may help simplify the demand for skilled workers in configuring and managing simple tasks, freeing them to spend more time on complex tasks that cannot yet be automated. This approach greatly simplifies the complexity of network management, lowers the management threshold, improves management efficiency, and at the same time, AI models can continuously learn and optimize their own algorithms, constantly improving the accuracy and efficiency of network management.

The emergence of intent-level templates makes network management more intelligent and efficient. With the help of intent-level templates, users can input their intent more freely, without worrying about whether their language cannot be recognized by the network or if they have missed some important information. It automatically translates user input intent through intent-level entities mining at both the word and semantic level. This technology has high accuracy and adaptability, and can adapt to different types and formats of user input.

Service-level templates and user-level templates are intermediate service templates. They are both network management technologies based on AI models, with higher flexibility and adaptability. Service-level templates determine template types according to user needs, such as QoS templates, ACL templates, etc. These templates allow users to control network behaviour at a finer granularity, thus better meeting user needs. For example, the QoS template can be used to set the priority of network traffic, thereby ensuring the smooth transmission of important business traffic. User-level templates are templates that correspond to user input. It is not only translated from the intent of user input, but also combined with factors such as user's preference habits, current network environment, etc., to give the best templates. This technology allows AI models to directly learn human preferences and habits, thus better meeting user needs.

6.1.2.3 Actors and Roles

There are three actors. Their roles are as follows:

- Regular user: an end-user of the system that has submitted and/or is using an intent policy.
- Network administrator: a person responsible for managing and maintaining a network.
- ENI System: the computer system that ingests, validates, and processes intent policies.

6.1.2.4 Initial context configuration

- Regular users input abstract intentions.
- AI models are launched and ready to be used.

6.1.2.5 Pre-conditions

In a multi-user environment, each user's QoE/QoS is guaranteed and not affected by competition for network resources.

6.1.2.6 Trigger conditions

- The change in user intent will cause the AI model to analyse and mine intent-level entities.
- The change in the network environment will cause the service- and user-level templates to be reconstructed.

6.1.2.7 Operational Flow of the Actions

- Users input abstract natural language intent to express their needs for network configuration. For example: "The online meeting of all students will be held at 9:00 today".
- The AI model combined with the external knowledge base performs entity mining on user intent to form key-value relationship pairs. At the same time, the AI model deeply understands the intent statement to form other intent-level entity information that is not explicitly required but necessary in the intent statement. For example: <Object>: all students, <Allow><Service>: online meeting, <Time>: 9:00.
- The AI model selects the corresponding service-level templates based on the intent-level entity type and predicted user intent, mainly including Quality of Service (QoS) template, Access Control List (ACL) template, For example, the QoS template is: <Object>: all students, <Service>: online meeting, <Time>: 9:00, <Bandwidth>:[20 Mbps, 100 Mbps].The ACL template is: <Src>: IP index of all students, <Dst>: IP index of the server, <Time>: 9:00, <Middlebox>:(firewall, admit).
- The system introduces an external knowledge graph to help establish a multi-user QoE/QoS model.

- Different strategies are adjusted for the detailed differences in different service-level templates, while user-level templates are established based on user preferences and the current network environment. For example, for QoS template User sub-template 1：:<Object>: IP index of student 1, <Service>: online meeting, <Time>: 9:00, <Bandwidth>:[25 Mbps, 100 Mbps]. User sub-template 2：<Object>: IP of student 2, <Service>: online meeting, <Time>: 9:00, <Bandwidth>:[30 Mbps, 120 Mbps]. User sub-template 3, etc.
- The system translates the user-level templates of different users by calling algorithms to generate network policies for each user.

6.1.2.8 Post-conditions

- The user's intent has been accurately understood and parsed, and has been translated into executable network policies.
- The network policies have been optimized based on the user's preferences and the current network environment.
- The network policies have been successfully issued to the corresponding network devices, such as firewalls, routers, etc.
- In a multi-user environment, each user's QoE/QoS is guaranteed and not affected by competition for network resources.
- If the network environment changes, the system can automatically adjust the network policy to maintain the best quality of service.

7 Summary and Next Steps

7.1 Open Issues for the present document

Void.

7.2 Issues for Future Study

From clause 4.3 (Functional Architecture):

- An updated workflow that can be used in a multi-domain environment.

NOTE: Domain infers to different network environment, such as core network, enterprise network, SD-WAN, etc.

From clause 6 (Use Cases):

- Additional use cases may be added to a future version of the present document.

History

Document history		
V4.1.1	August 2024	Publication