Network design system recommending similar configurations characterized by manual knowledges

Yuji KOJIMA, Kodo RAN, Yasuhiko AOKI, Takeshi NISHIYAMA
FUJITSU LIMITED
Kawasaki, Japan
{ kojima.yuuji, ran.kodo, aoki.yasu, nisiyama.takesi }@fujitsu.com

***Abstract*—Recently, the number of skilled network design engineers is not enough, and the improvement of non-expert's working efficiency and design quality by the design support system becomes an important issue. Therefore, we developed the network design support system that recommended similar configuration files when the engineer made new design. Our system converts the dependent part of each design matter in existing configuration files into the representation that showed design characteristics by use of the dictionary extracted from the manual of network device. It performs clustering (unsupervised learning) of the converted configuration files, and finds similar ones. We report the concept and implementation of the developed system.**

Keywords—Network Design, Configuration File, Clustering, Machine Learning

I. INTRODUCTION

Recently, the lack of the IT engineer is a serious social issue in Japan. Due to these situations, it becomes more and more difficult to secure skilled network design engineers every year. Therefore, it is an urgent necessity to promote handing down the skill from expert to non-expert, and to improve non-expert's working efficiency and design quality. The business process of network design consists of defining requirements, arranging a network layout, fixing an equipment composition, making a configuration file (config) and verifying a behavior of network.

Reference [1] reports that 49.8% (1st rank) of the respondent answer "Check the config" and 37.9% (2nd rank) of one answer "Make the config" as the work taking a lot of time in the setting change of the network (multiple answers allowed). Because of this, we expect that the support of making a config is useful to improve non-expert's working efficiency.

Further, reference [1] reports that CLI (Command Line Interface) is used in the case of about 50% of router/switch configurations, and GUI (Graphical User Interface) is used in the case of about 60% of firewall, load balancer, IDS/IPS configurations. In the future, a virtualization / standardization will proceed from the dedicated equipment in the router / switch, and the design by the use of GUI will probably be increased. Certainly, the intuitive GUI design can improve the work efficiency and design quality of non-experts. However, the text-based design remains deeply rooted at present. We think that a certain amount of functions which cannot be standardized remains, and expert's know-how which cannot be caught by standardization is reflected in the text-based configuration.

On the other hand, the documents of requirements definitions, configuration specifications and so on are written in natural language (Japanese, English, etc.) in many cases, and in order to realize practicable accuracy, the dictionary for correctly recognizing technical terminology in natural language is necessary. This dictionary has to be created manually over a long time, or the special technique (such as [2]) is required so as to automatically expand the words in the dictionary based on some manual works. Compared with a machine-readable configuration, when the system is developed based on a natural language document, there are concerns that technical hurdles may cause significant development costs.

Considering the effectiveness, persistence and costs as mentioned above, we decided to support non-expert's making the text-based configuration by our developed system.

The following Chapter 2 summarizes an overview of our developed system. Chapter 3 details the design concept of each function in the system. The conclusion is presented in Chapter 4.

II. OVERVIEW OF SYSTEM

The overview of our developed system is described in Fig. 1. The system recommends the commands and options that should be used for the configuration file to be designed when the network design engineer designs a new or updated network. At this time, the designer can obtain a recommended result from the system after inputting a draft version of the configuration into the system. The designer is required to create a draft version of the configuration in advance. However, this is not a problem. In actual design process, even if a new network is designed, there are few cases where its configuration is created from scratch, and in most cases, it is created from a similar model configuration. Therefore, inputting a draft version is more in line with the existing design process. Also, if the existing network is updated, it is so.

The process from the system's receipt of the draft to the recommendation is described below.

First, the area where the command and option is defined as shown in Fig. 2 is extracted from the manual of the network device by our developed tool, and the manual knowledge dictionary is created in advance (**dictionary formation**). The system can identify the commands and the options by the use of the manual knowledge dictionary.

Next, the system converts the dependent part of each design matter in a plurality of reliable configuration files with operational experience into the representation that showed design characteristics by use of the dictionary (**characterization**). One of the simple example of the dependent part is IP address. While the IP address "192.168.13.70" is not important, the action of setting IP address is important in terms of design characteristics. Therefore, the value "192.168.13.70" is converted into the

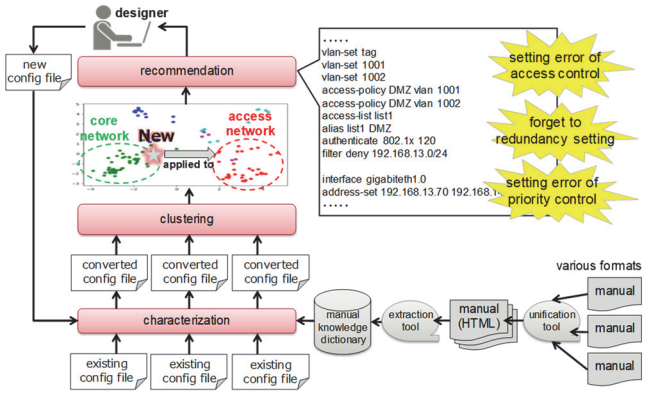

Fig. 1 Network design support system

term “src-IP-address”. The dictionary is required to support a large amount of various commands and options.

Next, the system performs the clustering (unsupervised learning) of a configuration group which consists of the existing configuration and the new draft version of the configuration that has been characterized, and classified it into the similar configuration group as shown in Fig. 1 (**clustering**). The designer basically selects the configuration group similar to the new configuration as the referenced configuration.

Finally, the system recommends the commands and options for the new configuration based on the status of use of the referenced configuration (**recommendation**). Concretely, when the command or option with a high utilization rate in the referenced configurations is not used in the new configuration, the system recommends the unused command or option.

A designer inputs the new configuration file, and then receives the recommendation result by accessing the network design recommendation server via the web browser client.

The system can prevent omitted settings, setting errors, useless settings and so on in making the configuration, and will improve the working efficiency and design quality of the network design.

In the following, the design concept of dictionary formation, characterization, clustering and recommendation described above are explained. Note that the commands and options in this paper are fictitious codes for explanation and have nothing to do with a particular network device.

III. Design Concept of System

A. *Dictionary formation*

The area where the command and option is defined as shown in Fig. 2 is extracted from the manual of the network device by our developed tool, and the manual knowledge dictionary is created in advance.

The network device manual is described in various document formats depending on the device vendor. The system coverts various document formats into HTML documents as shown in Fig. 1. A general conversion tool is used for the conversion.

Basically, the command and option are identified based on the tags of the HTML document and the notation in the area where the command and option is defined (e.g. SYNOPSIS, SYNTAX) as shown in Fig. 2.

The tags include, for example, paragraph <p> tag, boldface <b> tag, italic <i> tag, and the like. Depending on

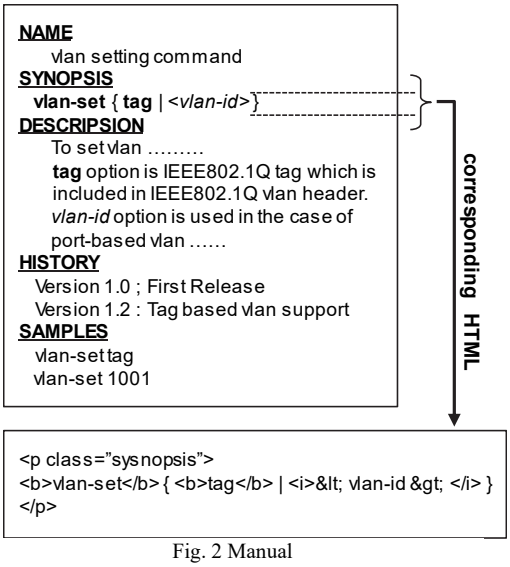

Fig. 2 Manual

the device vendor, the above definition area is located based on the paragraph <p> tag, the boldface <b> tag indicates the command or the option to input as it is, and the italic <i> tag indicates the option for the designer to input its value. In the manual, the option described as “ip-address” in italics is written as “192.168.13.70” in the actual configuration.

Regarding the notation, the symbol “[…]” indicates the selection of 0 or more options, the symbol “{…}” indicates the selection of 1 or more options, the symbol “…|…|…” indicates the selection of only 1 option, and the symbol “<…>” indicates the option for the designer to input its value.

Although there are subtle differences among device vendors, these typeface and notation are highly common. These are basically defined in the same way as SYNOPSIS displayed by the UNIX (Linux) man command. Therefore, the source code of the extraction tool for one vender can be easily diverted to the source code for another vender.

It is important to create the dictionary so that the system can identify differences commands and options supported depending on the difference in device model and operating system (OS) version. One of the most common mistakes when creating a configuration is caused by the difference between the device model / OS of the existing configuration referred by the new configuration and one of the new configuration. The system constructs the dictionary for each device model / OS and uses this knowledge when recommending.

Also, in dictionary formation, the high maintainability of the dictionary is important. At the time of the release of the new device model, there are many cases where the system becomes obsolete and will not be used because the maintenance cannot catch up. On the other hand, in our system, once the extraction tool for each device vendor is implemented, the dictionary can be automatically extracted from the manual thereafter.

B. *Characterization*

The system converts the dependent part of each design matter in a plurality of reliable configuration files with operational experience into the representation that showed design characteristics by use of the dictionary.

The dependent part of each design matter corresponds to the option for the designer to input its value. For example, it is interface-name, src/dst-IP-address, VLAN-ID, VPN-ID,

access-list-name, etc. What is important in characterization is at what degree of abstraction the option should be identified and can be identified. The following three aspects should be considered about this issue.

- Abstraction degree
- Over abstraction
- Alias / abbreviation

These aspects are explained below.

For example, it is assumed that the following is described in the manual and the configuration.

Manual:

vlan-set { **tag** | *<vlan-id>* }

Configuration:

vlan-set tag

vlan-set 1001

vlan-set 1002

The following three degrees can be considered as the abstraction degree of the option which the command "vlan-set" has.

	Value	→ **Object**	→ **Feature**
vlan-set	tag	→ N/A	→ tag
vlan-set	1001	→ vlan1001	→ vlan_id
vlan-set	1002	→ vlan1002	→ vlan_id

Regardless of what vlan-id value is set for each network design matter, if there are many cases where vlan-id option is set, the system should recommend to set vlan-id option. Therefore, when the system classifies the configurations into similar groups, the options should be abstracted at the feature level that represents the design characteristics such as the above.

However, vlan 1001 and vlan 1002 are different "objects" to which the system set. As described in Section D "Recommendation" below, if the command "access-policy" is well set to each "vlan-id" in addition to the command "vlan-set", the command "access-policy" should be recommended. In this case, if "access-policy" is set to vlan 1001 in the new configuration, but "access-policy" is not set to vlan 1002, then "access-policy" should be recommend only for vlan 1002. That is , it is necessary to identify vlan 1001 and vlan 1002 as different setting objects.

Given the above three abstraction degree as a basic framework, there are cases where the commands are defined in descriptions that are over-abstracted such as "method", "value", etc. in the manual. In this case, the simple conversion covers the design characteristics and make it invisible.

Manual:

authenticate *<method> <value>*

filter *<method> <value>*

The option "method" or "value" is the common term that many other commands can have, and is inappropriate to represent the design characteristics. In other to solve this issue, the extraction tool changes these options into the terms that represent the design characteristics and registers them in the dictionary. For example, it changes the option "method" into "802.1x", "EAP", etc. in the case of the command "authenticate". Its design characteristics appear at this degree of abstraction.

The alias or abbreviation "…" may be available in the manual or configuration.

Configuration:

vlan-set 1001 … 1010

access-list list1

alias list1 DMZ

The system is required to identify ten vlans of vlan 1001, vlan 1002, …, vlan 1010 as different objects. Also, the system is required to identify list1 and DMZ as the same object with the different face of access-list-name.

The system converts the configuration with different abstraction degree (value → object → feature) by the use of the manual knowledge dictionary, and uses the converted configuration for different purposes in accordance to the application (i.e, clustering, recommendation) as described below. The aliases and abbreviations are converted or decomposed according to the notation rules. On the other hand, regarding the over-abstraction, there are cases where it is not possible to obtain the appropriate abstracted representation without scanning the entire manual. Due to it, the dictionary has to be customized by some manual work. The challenge to reduce this work remains as the future study.

C. Clustering

The system performs the clustering (unsupervised learning) of a configuration group which consists of the existing configuration and the new draft version of the configuration that has been characterized with the abstraction degree of the feature level, and classified it into the similar configuration group as shown in Fig. 1.

The designer selects one configuration group similar to the new configuration, or another configuration group not similar to the new configuration but similar to the configuration of the network to which the new configuration is applied. For example, if the new draft version of configuration is created based on the existing configuration for the core network, but it is applied to the access network, the designer may select the cluster formed by the configurations for the access network as shown in Fig. 1.

In order to make such a selection, the system is required to manage the metadata (or attributes) of each configuration. The metadata includes, for example, a device model, an OS, a timestamp of last modification, an applied network type and the like. The designer finds the tendency of clusters by the use of metadata as a clue. In some cases, the designer perform the clustering again only for a particular cluster in order to narrow down the configuration to be referenced.

In the system, the process of clustering vectorizes the characterized configurations by the use of Doc2Vec [3] in configuration units, and clusters vector values by the use of K-means. Also, multi-dimensional vector values are degenerated into two dimensional ones by the use of PCA (principal component analysis) so as to visualize ones as the two-dimensional map as shown in Fig. 1. Simultaneously with clustering, the cosine similarities of the vector values of the

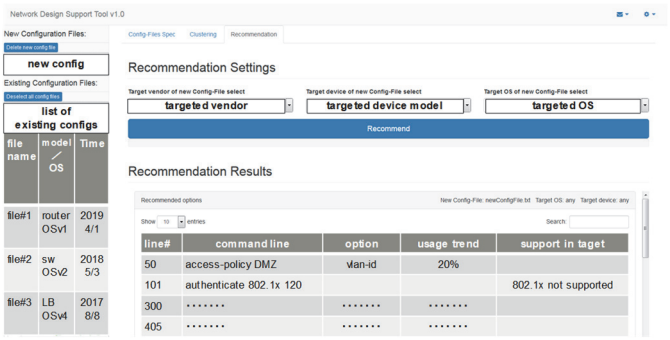

Fig. 3 Option recommendation per command

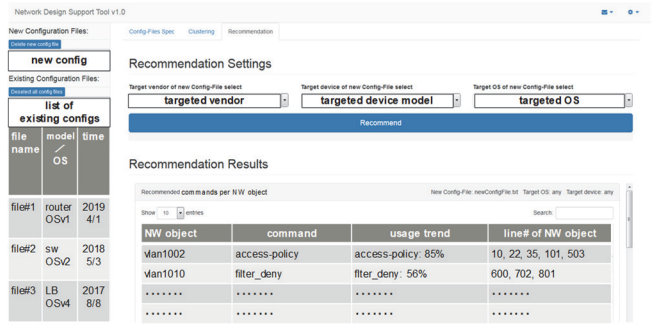

Fig. 4 Command recommendation per network object

existing configurations to one of the new configuration are calculated, and the list in order of the cosine similarity is showed.

Simultaneously with clustering, the cosine similarity of the vector value of the existing configuration to the vector value of the new configuration is obtained, and a list in order of the similarity is displayed.

D. Recommendation

The system recommends the commands and options for the new configuration based on the status of use of the referenced configuration which the designer selects considering the above result of clustering.

When obtaining the recommended result, the designer designates the device model and the OS version to which the new configuration is to be applied. In this way, the system recommends the commands and options in consideration of unsupported commands and options for the designated device model and OS version. That is, if new configuration contains the unsupported commands and options, the system warns the designer of it. The system thereby constructs the dictionary for each device model / OS on the above dictionary formation.

The commands and options are recommend from the following two views.

- Option recommendation per command
- Command recommendation per network object

Regarding the option recommendation per command, the system calculates the utilization rate of each option per command (e.g. vlan-set, authenticate) for the referenced configuration, and then if the command of the new configuration does not use the option whose utilization rate exceeds the threshold, the system recommends the corresponding command showed on the line where the corresponding command is set as shown in Fig. 3.

Regarding the command recommendation per network object, the system recommends the command which is not

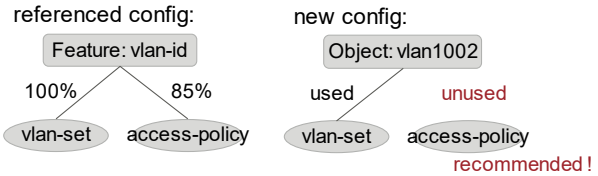

Fig. 5 Utilization in command recommendation per network object

used in the new configuration if the corresponding commands is well used over threshold for each object described in the above "characterization" in the referenced configuration as shown Fig. 4. For the example, the command vlan-set is set for the feature "vlan-id" at 100% utilization rate, and the command access-policy is set for it at 85% utilization rate as shown in Fig. 5. The threshold is 80%. The command vlan-set is set for the object "vlan1002" while the command access-policy is not set for it in the new configuration. In that case, the system recommends the command access-policy only for the object "vlan1002" except for the object "vlan1001".

At the time of recommendation, the system presents the usage trend of the command and option in the referenced configuration to the designer as optional information. By referring to the usage trend, the designer can find that the utilization rate is generally higher than the threshold for the recommended command and option, but the rate is low for the last few years of the referenced configurations. Although the command or option is grammatically supported within the designated device model and OS but not used recently, it may cause performance degradation, is scheduled to terminate the support. That is, there may be a reason to avoid it. This is an important indicator to reveal the invisible know-how of the expert.

IV. Conclusion

We developed the network design support system which recommends the commands and options that should be used for the configuration file to be designed when the network design engineer designs a new or updated network. The system converts the dependent part of each design matter in configurations into the representation that showed design characteristics by use of the dictionary extracted from the manual of network device. The system finds the similar configuration based on the clustering of the characterized configuration. The system makes various recommendations using different characterized configurations with different abstraction degrees.

The system aims to improve the work efficiency and design quality of non-experts.

References

[1] @IT, Juniper Networks, "The trouble that the network operator has in the survey of @IT readers," *(This original title and content is in japanese.)* https://www.juniper.net/assets/jp/jp/local/pdf/additional-resources/atmarkit-junos-survey-jp.pdf, May 2016.

[2] Takaya Yanagidaira, Kouki Sato, Satoshi Sunaga, Koji Hoshino, Kazuhiro Kikuma, Kiyoshi Ueda, "Improvement of compound word generation accuracy in automatic test item extraction method for large-scale software development," (*This original content is in japanese.*), IEICE Technical Report, vol. 118, no. 250, NS2018-112, pp. 39-42, Oct. 2018.

[3] Quoc V. Le, Tomas Mikolov, "Distributed Representations of Sentences and Documents," In Proceedings of the 31st International Conference on Machine Learning (ICML 2014), pages 1188-1196, Beijing, China, June 2014.