

# The Effects of Sharing Know-How Method Based on Experiences in Large-Scale Data Center

Hiroyuki Nishino and Mikifumi Shikida

**Abstract**—A new method for sharing know-how about “root cause analysis (RCA)” in large-scale data centers is proposed and evaluated. Recently, data centers are getting bigger because higher management efficiency is required. Moreover system managers are finding it ever-more difficult to understand the complete picture concerning the system. Consequently, RCA is likely to depend on the skilled or dedicated managers. The form of management contains a lot of disadvantages for stable management. The proposed method improves the situation concerning sharing the know-how possessed by skilled or dedicated managers. First of all this paper shows the background of data center management, second of all it shows that the proposal method with using relevant works, and finally it discusses about the effects of sharing know-how.

**Index Terms**—Root cause analysis (RCA), large-scale data center, know-how sharing, system dependency, system manager, technical training.

## I. INTRODUCTION

Recently, services provided by data centers are increasingly in demand. Moreover, data center must repeatedly eliminate and consolidate huge amounts of data. As a result, they are getting bigger and bigger. In addition, the number of system managers is decreasing. Due to these changes, many methods for root cause analysis have been proposed for the purpose of helping system managers. However, they do not enable unskilled managers to restore systems perfectly. Although they must ensure a stable operation of a system, the details of each system component is depend on each manager. On that account, in the present work, a method for sharing a skilled manager’s know-how based on their experiences and of preventing recurrence of the past system failures is proposed and evaluated.

In the rest of this paper, relevant works are described first. The proposed method for supporting system managers is then presented. After that, significant effects of the method on the performance of system managers are explained.

## II. RELEVANT WORKS

Relevant works on root cause analysis, system dependency, and know-how sharing are described in the following subsections.

Manuscript received September 2, 2015; revised December 10, 2015.

H. Nishino was with Japan Advanced Institute of Science and Technology, Japan (e-mail: hnishino@abeam.com).

M. Shikida is with Research Center for Advanced Computing Infrastructure, Japan Advanced Institute of Science and Technology, Japan (e-mail: shikida@jaist.ac.jp).

### A. Root Cause Analysis

A network-restoration method that uses not only event information about links that are down, but also the service error information was proposed by Miyazawa and Nishimura [1]. This method improves the accuracy of root cause analysis of networks by correlating plural information. However, a whole of server system contains too many possible root causes to correlate. A method of root cause analysis by correlating root causes and certain phenomenon was proposed by Nagai and Nagura [2]. The method proposed by Tonouchi [3] uses error messages in different systems with the same constitution. In addition, a proposed by Katoh and Shikida [4] estimates a root cause by using pattern-matching rules.

On the other hand, a method of correcting management information was proposed by Nagata and the others [5], and a method using network a management system (NMS) was proposed by Takahashi *et al.* [6]. However, all of these methods do not enable unskilled managers to restore errors in a huge system.

### B. System Dependency

A method of correcting the dependency of servers was proposed by Mori and Shikida [7]. This dependency can be applied to root cause analysis. Actually, the computational complexity of root cause analysis was reduced by Ikuse *et al.* [8] by using system dependency

### C. Methods of Know-How Sharing

On the other hand, there are many methods not to bring about any troubles. A method of systematically arranging information was proposed by Saitoh [9]. It also helps staff working in different sections, because the rules for information arrangement make information much easier to search. It is especially effective in the companies whose employees take many business trips or are often transferred. Moreover, a method of indicating some notes when an employee attempts to do an appropriate task was proposed by Shikida *et al.* [10]. Both methods are effective in preventing human error.

## III. METHOD OF KNOW-HOW SHARING

A method of preventing recurrence of failure in large-scale data centers is proposed in the following.

### A. Overview

Although methods of RCA can shorten the repair time, the managers cannot share the reason that particular error event happens. Accordingly, the proposed method retrieves error

events as “error risks” when managers are apt to execute operations. Unskilled managers are told the reason that a particular past error happened then. An overview of the proposed method is shown in Fig. 1. The method comprises the following processes.

- 1) Repairing work done in the past (Skilled managers)
- 2) Inserting know-how based on experiences in the past (skilled managers)
- 3) Operation (unskilled manager)
- 4) Retrieving error risks before execution

Each detailed process is explained separately as follows.

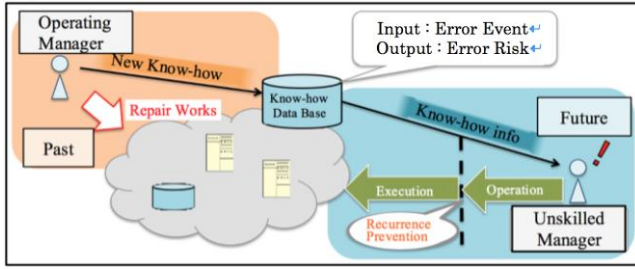


Fig. 1. Overview of proposal method.

### B. Inserting System Error as Know-How Information

This method treats skilled manager’s know-how as “know-how information” which consists of the following five items.

- Root-cause point

It is the output of RCA, and it is expressed by object. This information cannot be modified because the root-cause point needs to be specified.

ID : <Root-Cause Object>

- Root-cause operation

The operation becomes the start of errors in the operation tool’s record. Recently, a lot of operation tools are used by managers; therefore, they are distinguished by an “operation tool ID”. The root-cause operation is distinguished by the combination of operation tool ID and operation ID.

ID : <Operation Tool><Operation>

- Similarity-estimation range

The object group related with the error in root cause’s dependency. It is expressed by plural objects which are chosen by the conjecture of a system manager.

ID : <Object1>-<Object Type1>, <Object2>-<Object Type2>, <Object3>-<Object Type3>, ...

- Error reason

The reason that a system error occurred. It is written by a skilled manager.

ID : -

- Error-reason point

The point that contains the error reason. It is inserted by the object that has dependency. If the manager trying repair work does not know the error reason, it is taken as “0”. Other managers can edit this information when they understand the reason that the error occurs.

ID : [<Object> | 0]

Each item has an identifier (given in reference [7]). The objects used in the proposed method are listed in Table I. Every object has a plural “main type” and “sub type”. The retrieval of know-how information depends on the combination of these types.

### C. Retrieving Know-How Information

Both the case and the reason that a system error occurred by a certain operation and the reason may or may not follow a single pattern. Some operations may have gigantic know-how information. For that reason, know-how information should be rearranged when the managers execute the target operation. Accordingly, this method estimates the similarity rate of know-how information by using the ID stated above. The proposed algorithm for sorting Know-how Information is shown in Fig. 2.

TABLE I: OBJECTS ADOPTED AS KNOW-HOW INFORMATION

Class	Applicable Item	Main Type	Sub Type
Dir	Data on Memory	Host name Path	
VM Dir	Data on Virtual	Host name Path	
Disk	Physical Memory	Host name Disk name	Vendor Product
VM Disk	Virtual Memory	Host name Disk name	Hypervisor Version
Network	Protocol	MAC Address	Protocol Hop count

```

1 : foreach()
2 : {
3 :   if( "Similarity Estimation Range" contains "the Error Reason Point" &&
4 :       corresponding object type of "the Error Reason Part" is
5 :       different from know-how information )
6 :     ( "Similarity Rate" ) = 0;
7 :   else
8 :   {
9 :     foreach( All of object type ID in "Similarity Estimation Range" )
10 :    {
11 :      if( object ID == The corresponding object of "Target Operation" )
12 :        ( Consistency Rate ) ++;
13 :      else
14 :      {
15 :        ( weight of object type ) = 0.9 / ( the number of object type )
16 :        foreach(object type of each object )
17 :        {
18 :          if( object type ID == object type ID of corresponding object )
19 :            ( Consistency Rate ) += ( weight of object type );
20 :          else if( Network object )
21 :            continue;
22 :          else
23 :            break;
24 :        }
25 :        if( Dir object || VMDir object )
26 :          ( Consistency Rate ) += 0.45;
27 :      }
28 :    }
29 :    ( Similarity Rate ) = ( Consistency Rate ) / ( the number of objects in
30 :      "Similarity Estimation Range" )
31 :  }
32 : }

```

Fig. 2. Algorithm for calculating similarity rate.

## IV. DISCUSSION

### A. Authentic Account

This method is especially helpful for the different systems that have similar constructions. However, just because the constructions are similar does not necessary mean the same know-how information is applicable.

This chapter examined about the authentic account in the school where the author belongs. In this case study, trouble with breakdown in the system occurred. The constitution of the system is shown in Fig. 3.

Each laboratory uses the servers, named “lss3” in the figure. The servers are running on VM ware, and the VMware is installed on remote disks configured as a plural disk array. Typically, system errors are caused by a certain operations. The operation was applied to the network (VLAN) between the disk array and the virtual machine. Then, all hosts cannot access to the lss3 servers at once. Although the operation is

appropriate, a system error occurred. The reason for the error is the bad combination between the operation and the constitution. Changing the configuration of the VLAN can cause of a system error when there are some switches in the appropriate VLAN. In that case, the root-cause point (lss3) and the error-reason point (switch) are different. The error depends on the existence of a switch in the similarity-estimation range. The proposed method estimates the possibility of causing an error pattern when the applicable operation is likely to be executed. In addition, it is also able to execute the know-how information likely to happen then.

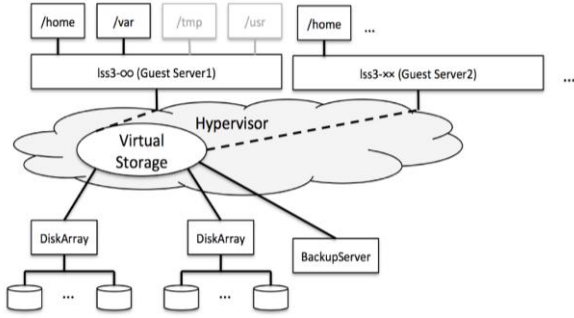


Fig. 3. Real example in author's school.

The effect of the proposed method was evaluated by comparing the amount of stored know-how information and comparing different systems with similar configurations.

### B. Usability of the System

In the large-scale data centers, there are gigantic objects, and almost all of which have a lot of object types, exist. The dependency of what on what is too complex for managers to understand the complete picture. Accordingly, the proposed method informs managers of the risks of system error without understanding the complete picture just before they execute a certain operation.

The usability of the proposed method was verified in the example concerning know-how information with a simple similarity-estimation range (Table II). The detailed premises are listed below.

- The know-how information has three objects (disk, VM disk, and network) in the similarity-estimation range.
- Each object has two object types.
- A large amount of know-how information has already been stored in the know-how database.

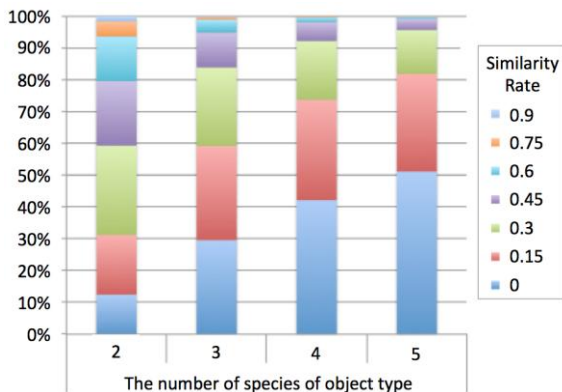


Fig. 4. Comparison of indication ratios.

The number of species of each object type has a significant effect on the number of know-how information patterns. The

result of the verification of the algorithm when its quantity increases two to five are shown in Fig. 4.

TABLE II: OBJECTS IN SIMILARITY-ESTIMATION RANGE

Object	Disk		VM Disk	
Object Type	Manufacturer	Product	Hypervisor	Version
Species	HGST	Mega Scale	ESXi	5.5

Object	Network	
Object Type	Protocol	Hop-Count
Species	NFS	1

The bar graph shows that the more the number of object types, the lower the ratio of know-how information with high similarity rate. In addition, The amount of know-how information that is not indicated (Similarity rate: 0) increases with increasing number of object types. In other words, the increase of the amount of know-how information is not a big deal.

As a result, the proposed method can estimate the know-how information that contains high similarity rate in the target system even if the operation is on a long term basis. Accordingly, the indicated information is beneficial one only.

### C. Supporting System Management

Fixing complex system error in the important systems usually depends on skilled managers. This trend is no good for the education of unskilled managers. Accordingly, the proposed method archives the know-how of skilled managers as know-how information, and show it to unskilled managers when the operation has been executed. Consequently, they can gain experiences of cases that they have not experienced before even though they have not accumulated enough careers. To consider if system errors occur by using know-how information is good for their education.

## V. CONCLUSION

A method of informing in experienced managers of past system errors as “know-how information” was proposed and evaluated. The configuration of data centers, from each machine to each service, is becoming more complicated. As a result, system managers are getting flustered. However, this proposed method makes troubleshooting much easier. In addition, it can not only prevent system errors before anything they occur, but also educate new system managers to display know-how information before each operation.

The proposed method significantly improves the degree of similarity finer, or to estimate unknown system dependency by using a hidden Markov model. Henceforth, it will be introduced to not only data centers but also many companies managing huge systems.

## REFERENCES

- [1] M. Miyazawa and K. Nishimura, “Proposal of method for root cause analysis designed for service quality management,” *IEICE Technical Report on Information and Communication Management*, vol. 110, no. 466, pp. 7-10, 2011.
- [2] T. Nagai and M. Nagura, “Root-cause analysis system for rapid hazard recovery on large-scale environment,” *Journal of Information Processing*, vol. 54, no. 3, pp. 1109-1119, 2013.

- [3] T. Tonouchi and M. Murata, "Root cause analysis technique for derivative failures with implicit dependencies," *IEICE Transactions on Communications*, vol. J92-B, no. 8, pp. 1236-1244, 2009.
- [4] Y. Katoh and M. Shikida, "Presentation method of optimal avoiding way of failure prediction," in *Proc. Internet and Operation Technology Symposium*, 2012, pp. 110-116.
- [5] T. Nagata, Y. Taniguchi, and S. Tamaki, "A proposal of large scale and distributed network management system," *IPSJ Technical Report DSM*, no. 113, pp. 31-36, 2000.
- [6] Y. Takahashi, D. Misugi, A. Takahashi, K. Sasai, T. Abe, and T. Kinoshita, "Network fault management support method by organization of activated knowledge," *IPSJ Technical Report CSEC*, no. 5, pp. 1-8, 2010.
- [7] H. Mori and M. Shikida, "The method of supporting system configuration management for using the dependency of a server," *Journal of Information Processing*, vol. 46, no. 4, pp. 940-948, 2005.
- [8] T. Ikuse, M. Enokimoto, H. Hazeyama, Y. Kadobayashi, and H. Yamaguchi, "Reducing computational costs of root cause analysis using dynamic dependency graph," *IPSJ Technical Report OS*, pp. 1-8, 2011.
- [9] N. Saitoh, "The knowledge sharing and inheriting method for company activities," *IPSJ Technical Report GN*, pp. 1-6, 2010.
- [10] M. Shikida, C. Kadowaki, and S. Kunifuji, "A proposal of an informal information sharing method by linking to flow in group work

processes," *Journal of Information Processing*, vol. 41, no. 10, pp. 2731-2741, 2000.



**Mikifumi Shikida** received a Ph.D. in engineering from Tokyo Institute of Technology in 1995. He is a professor at Research Center for Advanced Computing Infrastructure, Japan Advanced Institute of Science and Technology, Japan. His research interests are management methods for large-scale systems, groupware, and network services.



**Hiroyuki Nishino** received a master's degree in information science from Japan Advanced Institute of Science and Technology in 2014. He was in Internet Initiative Japan Inc., Japan. He is now with ABeam Consulting Ltd., Japan. His research interests are management methods for large-scale systems, TCP/IP network, and big data.