# Learning-Based Adaptation Determination Method for Problem Recognition of Self-Adaptive Software

**Kwangsoo Seol[1], and Doo-Kwon Baik[2]**
[1]Department of Computer and Radio Communications Engineering,
Korea University, Seoul, Republic of Korea

***Abstract*— In this paper, we propose a method for identifying the adaptation period when a problem occurs in a system in order to reduce the unnecessary adaptation of self-adaptive software. Consequently, the dangerous situation information is defined, the behavior information at the time of problem occurrence is learned, and the adaptive performance is determined by comparing it with the existing similar situations by using the k-nearest neighbors algorithm. By the use of the proposed method, a situation where an unnecessary adaptation process is performed while running the self-adaptive system could be avoided, system load may be reduced, and service quality may be enhanced.**

**Keywords—Self-adaptive software; Machine learning; Problem recognition**

## I. INTRODUCTION (*HEADING 1*)

Self-adaptive software-related studies have been conducted for a long time, but a number of current self-adaptive software studies follow a reactive method[1]. In the case where real-world information is collected through a sensor to detect the change in the state of reactive self-adaptive software, and the current state is determined to have a value over a threshold defined within a system, the state is judged to be a dangerous situation and an adaptation is performed. Because such an existing adaptation method determines whether to perform an adaptation without considering the situation that might change in the future, the problem of conducting unnecessary adaptation occurs. If an unnecessary adaptation process is repeated, the system load becomes severe and the service quality declines.

In this paper, we propose a learning-based adaptation period determination method by using a kNN algorithm in order to determine whether to execute an adaptation when it is determined that a problem has occurred in self-adaptive software. Through the proposed method, the system reduces unnecessary adaptation processes with respect to a problem that occurs during the operation, and conducts only the necessary adaptations, thereby cutting the cost required for conducting adaptations.

## II. RELATED WORK

MAPE-K, proposed by IBM, is deemed to be representative research on the self-adaptive system model [2]. MAPE-K is a feedback control loop consisting of four-step classes for conducting an adaptation. Each class is composed of the monitoring (M), analyzing (A), planning (P), and executing (E) steps, and the abbreviation MAPE is formed by taking the first letter of each step. In the monitoring step, surrounding environmental information is collected; in the analyzing step, the matter with respect to whether a problem has occurred is determined on the basis of the collected information. If a problem has occurred, an adequate adaptation strategy is identified in the planning step. Finally, the executing step is where the adaptation strategy is applied. “K” in MAPE-K refers to knowledge (K) information required in conducting the above steps. Recently, a number of studies on self-adaptive systems have used the MAPE-K model.

## III. PROPOSED METHOD

The system collects state information in real-time through a sensor on the basis of the monitoring elements defined in the design phase during runtime, and analyzes whether the current state has a problem. When analyzing the problem, the existing self-adaptive system compares the value of the collected information with the threshold value. In reality, however, a situation where an adaptation is unnecessary even if the collected value exceeds the threshold may occur.

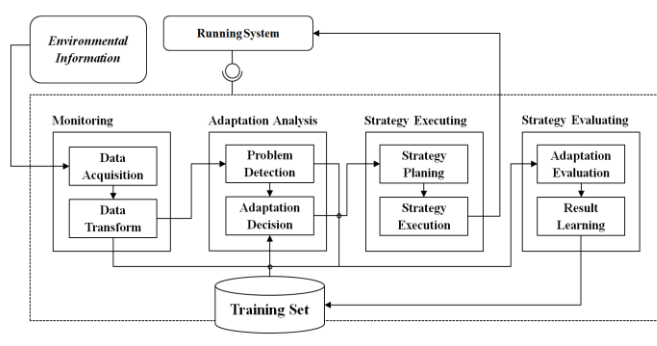


**Fig. 1** Schematic representation of the proposed system

In order to overcome such a problem, a method of determining an adaptation period using the probability based on learning is proposed in this paper. The proposed system is composed of a total of four steps as shown in Fig. 1.

### *A. Definition of dangerous situation information*

In the proposed method, a process of determining the learning elements for recording dangerous situation information is required. *Dangerous situation information* is used for finding a similarity between the collected environmental information and the existing training set within the process of determining whether the period when the value of collected environmental information has exceeded the threshold and has led to a dangerous situation is an adaptation period; the *dangerous situation information* is defined in the system design phase. *Dangerous situation information* defines the information on the change in the value obtained until the collected data indicate the dangerous situation. The related example is as follows:

*(1) Mean gradient from previous t seconds to dangerous level*

*(2) Time consumed until the previous dangerous situation*

*(3) Frequency of danger occurrence for t seconds*

Each learning element refers to the behavior information value right before the collected value reaches the dangerous level. The elements defined in this way are additionally required to be verified whether they are significant in the inference through actual learning. To this end, a cross-validation should be conducted after designing the system.

Because each learning element $F_i$ might be distributed in various numerical values, a normalization process is required to measure similarities by using F.

$NF_i$ refers to the normalized value and has values between 0 and 1. The $n^{\text{th}}$ dangerous situation information R designed as above can be defined in the following form:

$$R_n = < NF_1, NF_2, \dots, NF_m > \quad (2)$$

If the values collected in the monitoring step exceed the threshold, each normalized learning element ( $NF_i$ ) is calculated to generate $R_n$.

### *B. Dangerous situation analysis*

If the collected data exceed the threshold and lead to a dangerous step, the system executes the adaptation analysis step. In this step, the dangerous situation information $R_n$ and the learned data are checked to find situations similar to the current situation, and the matter with regards to executing an adaptation is determined through the statistics of the result values recorded in the relevant situations.

To this end, the similarity between the learned data and the current situation is measured by using the kNN algorithm and the Euclidean distance measurement method. The similarity measurement equation is as follows: $NF_i$ refers to the "i" normalized learning element of the current dangerous situation information and $sNF_i$ refers to the "i" normalized learning element stored in the learning data.

$$sim(x, y) = \sqrt{\sum_{i=1}^{n} (NF_i - sNF_i)^2} \quad (3)$$

Based on the measured similarity, the mean value $(mv)$ of result values $(D)$ having k top values is measured. The result values have the value of 0 or 1, and the process of generating the result values is described in Section 3.3. Then, the system determines whether to execute an adaptation by comparing the finally calculated mean value with the reference adaptation value $(rv)$ defined by the user. If $mv$ is larger than $rv$, the adaptation is executed, and in the opposite case, the adaptation is not executed.

### *C. Evaluation of adaptation result*

The step of evaluating the adaptation of the proposed method is the step of creating learning data. In this step, the matter with respect to whether the system should have or should not have executed an actual adaptation of the result t seconds after a dangerous step is recorded and stored as learning data on the basis of the actual data, not a prediction. The result value $D$ is 1 if an adaptation is necessary and 0 if it is not. The learning data format $sR_n$ stored in the learning storage is as follows:

$$sR_n = < NF_1, NF_2, \dots, NF_m, D > \quad (4)$$

## IV. Conclusion

In this paper, we proposed a method for determining the kNN learning-based adaptation period in order to execute an effective adaptation of self-adaptive software. Through this method, the execution of an adaptation can be determined on the basis of not only the current situation but also multi-dimensional elements. The proposed method can contribute to reducing the cost incurred in an adaptation by avoiding the execution of an unnecessary adaptation while running the system.

## V. Acknowledgement

This research was supported by Next-Generation Information Computing Development Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (NRF-2012M3C4A7033346). Doo-Kwon Baik is corresponding author

## VI. References

[1] M Salehie, L Tahvildari. "Self-adaptive software: Landscape and research challenges", Journal of ACM Transactions on Autonomous and Adaptive Systems (TAAS) Vol. 4 no. 2, 2009

[2] Kephart, J.O., and Chess, D.M. "The Vision of Autonomic Computing", Computer. Vol. 36 no.1, 41-50, 2003