ISSN (Print) : 0974-6846
Indian Journal of Science and Technology, *Vol 8(18), DOI: 10.17485/ijst/2015/v8i18/83956, August 2015* ISSN (Online) : 0974-5645

Generating System Knowledge for Autonomic Control

Jeongmin Park[1], Taebok Yoon[2] and Seunghwa Lee[3*]

[1]Department of Computer Engineering, Korea Polytechnic University, Siheung, 429-793, Korea; jmpark@kpu.ac.kr
[2]Department of Computer Software, Seoil University, Seoul, 131-702, Korea; tbyoon@seoil.ac.kr
[3]Division of Information and Communication, Baekseok University, Cheonan, 330-704, Korea; sh.lee@bu.ac.kr

Abstract

Recently, autonomic computing researches have been applied based on the concepts of self-adaptation. Traditional autonomic control methods are used to model the process or to maximize the system resources in the external view of the managed system. However, they produce a heavy workload for the self-adaption engineers that must analyze the external environment without having access to internal information to model the managed system. To solve these problems, it is highly recommended to specify the managed object by generating the system's knowledge. Thus, this paper describes an approach to implementing how to monitor system knowledge for autonomic control system. Through these activities, it is possible to minimize the efforts of analyzing the target system of a developer and reduce the system setting overloads for automatic setting of monitoring environment by obtaining the monitoring scope based on the internal system information. The approach is addressed in regard to defining system knowledge required for Autonomic Control from UML model.

Keywords: Autonomic Control System, High Reliability, Monitoring Knowledge

1. Introduction

As management complexity and maintenance cost of autonomic control system keep spiraling upward, it is a critical and difficult job for humans to maintain and manage autonomic system in a complicated computing environment. Thus, to realize an autonomic control system, the next following issues should be settled.

- **Monitoring:** It monitors the process of software. The subject of monitoring can be processing environment.
- **Analysis:** It analyzes the problems of software and evaluates whether the software needs autonomic control according to the importance of the problem.
- **Planning:** It seeks the autonomic control strategy to solve the problem.
- **Execution:** It dynamically changes the action and structure of processing system under the autonomic control strategy.

First of all, it is the most crucial and difficult to implement monitoring among these issues above. As this field of study is the early state of research, there are not many applications. However, in large there are earlier studies about autonomic control which can be classified as self-healing based on the model for external environment, the model for component, and self-healing based on log. The common problem is that the experts of self-healing have to examine essential features of target system directly. For instance, in order to analyze the resource environment of target system, component state model should be analyzed for the modeling constraints and analyzing internal state. All these efforts of the analysis vary on the degree of understanding of self-healing developer.

Thus, this paper suggests 'Towards Monitoring System Knowledge for Autonomic Control of CPS'.

* *Author for correspondence*

Through the suggestions, it is possible to minimize the efforts of analyzing the target system of a developer and reduce the system setting overloads for automatic setting of monitoring environment by obtaining the monitoring scope based on the internal system information.

2. Related Work

Recently, autonomic computing researches have been applied based on the concepts of autonomic control.

This is the most traditional autonomic control method[1,2], developed at Carnegie Mellon University in the U.S.A, and it suggested using a layered structural analysis model to observe, analyze, plan, and execute reconfiguration strategies. Thisapproachis used to model the process or to maximize the system resources in the external view of the managed system. As for its advantages, it allows for the development of an autonomic control system without an internal information system by not modifying the system. However, it produces a heavy workload for the self-healing engineers that must analyze the external environment without having access to internal information to model the managed system. To solve these problems, it is highly recommended to specify the managed object by modeling the system's goals.

In our previous studies[3], we suggested a self-adaptive system identifying internal states and the external resource environment using a UML model created in a software architecture stage. It detects an abnormal state of the external resources that is based on resource specifications in the deployment model, and it inserts the detected code converted according to XMI information analyzed in terms of class, sequence, state, and the model is run to detect abnormal actions inside the components. An advantage is that it can automatically create constraints for the internal and external state using amodel designed for the target system in order to reduce the effort necessary for analysis by the engineers who realize autonomic control software. Although it can identify an internal state and the component's execution environment according to the XMI information created in the designed models, it cannot identify a communication fault or an interaction fault between components, and therefore, the most general usable ranges are very limited. To solve these problems, more advanced autonomic control method is needed.

In the field of monitoring research, a study generating monitor based on the formally described system information has been actively progressed. The information in the system is used in a wide range of rules, constraints, the goal graph, fault trees, etc[4]. However, the system information used in studies of monitoring should also be analyzed manually similar to the existing autonomic control methodologies. Since the information modeling of system also uses 'formal method', specification of knowledge is difficult.

A common problem of previous researches is that it requires considerable costs for the additional task for system modeling, and environment construction for monitoring system toward autonomic control[5,6]. Thus, this paper aims to generate monitor knowledge in order to partially resolve the common vulnerabilities of the existing studies.

3. Monitoring Knowledge for Autonomic Control

In component based system, system knowledge has some information about components. It is required to enable monitoring of source code level and defines classes composing component and state information. Also it offers not only component state of accurate intention but also information about actually embodied class. System knowledge, SK defines the component's set, Comset and relation among components, Rset.

SK = {Comset, Rset}

SK includes not only information about the composition and the name of components but also that about relation among components. By way of this, when a problem event occurs in a particular component, it can offer information about its effect on the whole system. Rset includes ClassInfo (Class's Information), Stateinfo (Status's information).

Comi = {ClassInfo, State Info}

ClassInfo = {Clsset, ClaRelset}

Cls = {classname, Atrset, Operset}

ClassInfo is composed of Clset (Class set) and ClaRelset (Class's Relation set). Clsconsits of classname and Atrset (Attribute set) and Operset (Operation set). Here class name, attribute and function name are required to identify monitoring object by analyzing embodied code and makes possible detailed monitoring such as class generation, change of attribute and functional performance.

StateInfo = {Sset, Tset}

S ← {statename, Tin, Tout}

T ← {transname, statesour, state dest}

StateInfo (State information) consists of Sset (State set) and Tset (State Transition set). S (State) consists of state name, Tin (state's entry pointe) and Tout (state's exit point). T (state Transition) consists of transname(transition name), statesour (source state), statedest (destination state). These state information offers information about the state of a particular component when a system error occurs and also is used to extract monitoring rule as seen in our previous work[5]. T(state Transfer) assumed as functions in systems. Consequently, since monitor is based on the system knowledge including the information, it is possible monitoring internal system at the level of source code.

4. Case Study

4.1 System Modeling Stage

A Designer designs ATM and Server components as Figure 1 through component diagram of whole system.

Figure 1. Components of Target System.

Because ATM and Server are connected each other, they are perceived as connected components. Then design information about attributes and actions in each class of each component through each ATM class diagram and server diagram as Figure 2.

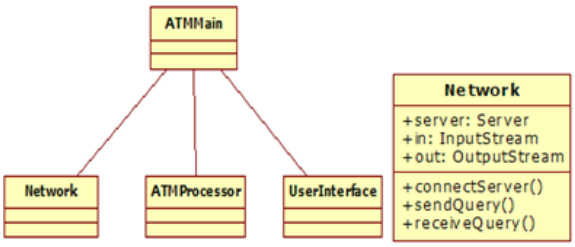

Figure 2. Class Diagram of ATM Components.

After finishing designing class diagram, design State diagram of each class diagram as Figure 3. The transition included in the state diagram is either the function in the class diagram or the automatic transition. After UML system model is described, it is transferred to XMI, and input in 'Monitor Generator'.

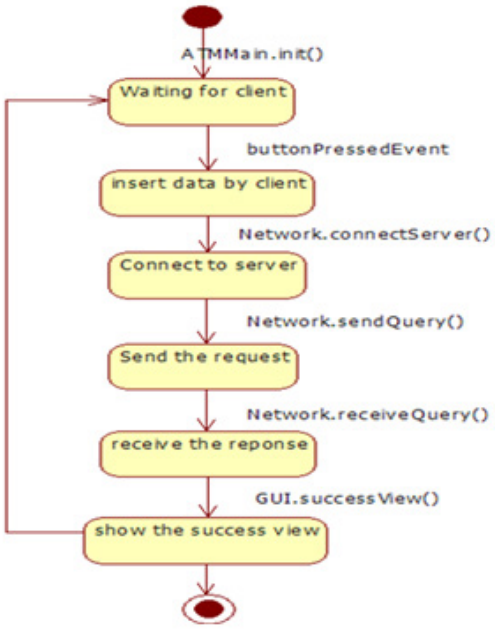

Figure 3. State Diagram of ATM Main Network.

4.2 Generation System Knowledge

When XMI file is input, 'Input Analyzer' analyzes it and 'System Knowledge Generator' reads ATM and Server components that consist of system and produces Comset.

Comset= {ATM, Server}

And it derives each of the class information from class diagram modeling each component. Thus information is derived from the exercise as follows:

ATMClassInfo= { Cls_{set} , $ClsRel_{set}$}

Cls_{set} = { ATMMain, Network, ATMProcessor , User Interface}

$ClsRel_{set}$ = {Rel_1, Rel_2, Rel_3 }

$ClsRel_{set}$ represents the association between classes. **Rel_1** represents the relationship between Network and **ATMMain**.

Rel_1 = {Association, Network, ATMMain}

Information about **ATMMain** class of ATM component is derived as follows:

ATMMain= { Atr_{set} , $Oper_{set}$ }

Atr_{set} = { server, in, out}

$Oper_{set}$ = {connect Server, send Query, receive Query}

Finally, state information of each component is derived. See only State $\mathbf{S_4}$ and Transfer $\mathbf{T_4}$, here. State $\mathbf{S_4}$ and Transfer $\mathbf{T_4}$ are described as follows:

StateInfo = { $\mathbf{State_{set}}$, $\mathbf{Trans_{set}}$ }

$\mathbf{State_{set}}$ = { **Int,** $\mathbf{S_1}$, $\mathbf{S_2}$, $\mathbf{S_3}$, $\mathbf{S_4}$, $\mathbf{S_5}$, $\mathbf{S_6}$, **End**}

$\mathbf{Trans_{set}}$ = {$\mathbf{T_1}$, $\mathbf{T_2}$, $\mathbf{T_3}$, $\mathbf{T_4}$, $\mathbf{T_5}$, $\mathbf{T_6}$, $\mathbf{T_7}$}

$\mathbf{S_4}$= { **Send the request,** $\mathbf{T_4}$, $\mathbf{T_5}$}

$\mathbf{T_4}$= { **Network Send Query,** $\mathbf{S_4}$, $\mathbf{S_5}$}

5. Conclusion

The thesis proposed an approach in order to decrease the burden of system analysis and monitor generation required previously by automatically generating component monitor within target system using UML design model of target system.

After confirming the process of proposal approach through case study about simple ATM system, the decrease of the burden of system analysis and monitor generation was confirmed. This thesis is a beginning study for automation of autonomic control system and makes it a final goal to generate an autonomic control system which was not available in the field so far due to the burden of development.

For further study, it is aimed to keep improving the range and the field of monitoring by way of generating a variety of monitoring rules and tactics through limitation technology using OCL and UML model. Also, various evaluation for adapting the existing framework in the field will be proceed.

6. References

1. Garlan D, Cheng S, Huang A, Schmerl B, Steenkiste P. Rainbow: Architecture-based self-adaptation with reusable infrastructure. IEEE Computer. 2004 Oct; 37(10):46–54.
2. Garlan D, Schmerl B. Model-based adaptation for self-healing systems. In WOSS 2002: Proceedings of the first workshop on Self-healing systems; 2002. p. 27–32.
3. Park J, Youn H, Lee E. An autonomic code generation for self-healing. Journal of Information Science and Engineering. 2009; 25(6):1753–81.
4. Kitamura Y, Ueda M, Ikeda M, Kobori S, Kakusho O, Mizoguchi R. A model-based diagnosis with fault event models. Proceedings of Pacific Asian Conference on Expert Systems; 1997. p. 322–9.
5. Park J, Chun I, Lee H, Kim W, Lee E. Intelligent service robot and application operating in cyber-physical environment. Lecture Notes in Electrical Engineering. 2012; 181:301–10.
6. Chun I, Kim J, Lee H, Kim W, Park S, Lee E. Faults and adaptation policy modeling method for self-adaptive robots. Communications in Computer and Information Science. 2011; 150:156–64.