Journal of Robotics, Networks and Artificial Life, Vol. 1, No. 2 (September 2014), 120-124

Proposal of a Method to Build Markov Chain Usage Model from UML Diagrams for Communication Delay Testing in Distributed Systems

Tetsuro Katayama[*†], Zhijia Zhao[†], Yoshihiro Kita[‡], Hisaaki Yamaba[†] and Naonobu Okazaki[†]
[†]*Faculty of Engineering, University of Miyazaki, Miyazaki 889-2192, Japan*
[‡]*Security Center, Kanagawa Institute of Technology, Kanagawa 243-0292, Japan*

[*]Corresponding author, E-mail: kat@cs.miyazaki-u.ac.jp
Tel: +81-985-58-7586, Fax: +81-985-58-7586

Abstract

As the growth of network technology with high parallelism and high reliability of distributed systems, they have been widely adopted in the enterprise and society. But, how to design for testing the real-time or a communication delay of distributed systems is not been discussed much. This paper proposes a new method to automatically build Markov Chain Usage Model. The proposed method establishes the time points and builds a new UML diagram which can test the communication delay of a distributed system to improve its reliability. We have confirmed the usefulness of the proposed method to adapt it to practical examples.

Keywords: distributed systems, communication delay, Markov Chain Usage Model, UML diagram, reliability

1. Introduction

Software testing is one of the key technologies to ensure software reliability. For decades, many software testing methods have been proposed. Building the software usage model becomes the basis of reliability evaluation and reliability testing. Software testing with Markov Chain Usage Model is an effective method to be used by programmers and testers in web sites development, to guarantee the software reliability.

Because UML (Unified Modeling Language) [1] has become a practical standard modeling language for object-oriented program, many software developers adopt UML diagrams for object-oriented development. How to build the usage model from UML diagrams has been proposed [2]. But, how to build a usage model from UML diagrams corresponding to the communication delay of distributed systems has not been discussed much.

In this paper, we aim to improve the reliability of distributed systems by testing the communication delay. To achieve this aim, we propose a new method to automatically build Markov Chain Usage Model by establishing the time points and building a new UML diagram. Specifically, we establish the time points and build a new UML diagram called De-sequence diagram based on deployment diagram and sequence diagram. Then, we propose an algorithm to generate the Markov Chain Usage Model from the new UML diagram.

2. Method of Building Markov Chain Usage Model

2.1 De-sequence diagram

For testing the communication delay of a distributed system from UML diagram, we need to describe interactive time and sequence between objects of system that distributed on each node. Sequence diagram shows the dynamic partnership between objects, but it does not

shows the layout of the distributed system, and it also does not describe the time of messages sent. For this reason, we combine it with deployment diagram, and set the time point to describe interactive time and sequence between objects. We call this diagram as De-sequence diagram.

As shown in Fig.1, De-sequence diagram puts the system distributed on each node as an object, and puts the life-line between time points as a state. It can describe interactive time and sequence between objects while describing dependencies of system distributed on each node.

Specifically, we show the definitions as follows:

Definition1. Let system that we will test as: Sys = ($Inst_1$…$Inst_n$), Sys is the system, and Inst is the instance.

Definition2. Let the instance of the system as: Inst = (Na, Node, TrPre, TrPost, Trans, Pf), Na is the name of instance, Node is the node that instance belongs to, TrPre is the Pre-condition of Transition, TrPost is the Post-condition of Transition, Trans is the Transition of the system, and Pf is the probability of the Transition.

Definition3. Let message of the system as: Mes = (Mna, Sour, Targ), Mna is the name of the message, Sour is the source instance of the message, and Targ is the target instance of the message.

Before the other definitions, we propose a method as follows to divide the system into some cases for establishing the time points easily in Definition5:

Method1. Find a message from the beginning. If the source instance of the message is the same as the target instance of the next message of the first one, and the target instance of first message is the same as the source instance of the second message, we call the message and the next message pair messages. Otherwise, we call the message single message. We construct a case with two instances and pair messages or single message. Construct the other cases as above.

Definition4. Let the case we constructed in method1 as C = (C_i, Mes, Inst, CPre, CPost), C_i is the number of the case, CPre is the Pre-condition of the case, CPost is the Post-condition of the case, we use CPre and CPost to describe the sequence of cases, Mes and Inst has been defined in definition2 and definition3.

Definition5. If a message let the case have a transition, we call this message CPre, the Pre-condition of the case. If a message be sent just after the transition of the case, we call this message CPost, the Pre-condition of the case.

Definition6. Let Pre-condition of Transition as: TrPre = (Tm, Tv, La), Tm is the time mark, Tv is time violation, and La is the label of system.

Definition7. Let time mark as: Tm = (Mna, C_it_j), Mna is the name of message, and C_it_j is the count of time points.

Definition8. Let time violation as: Tv = (Tc, TvH, TvPos, Tvpro), Tc is the time constraint equation, TvH is the time violation processing, TvPos is the Post-condition of time violation, and TvPro is the probability of time violation.

Definition9. Let system label as: La = (Initial, Middle, TiVio, Final), Initial is the initial stage of system, Middle is the intermediate stage of system, TiVio is the time violation stage of system, and Final is the final stage of system.

Definition10. Let Post-condition of Transition as: TrPost = (Tm').

Definition11. Let Trans as Transitions, it is a process to satisfy the TrPost on you are satisfied with the TrPre.

2.2 Definition of Markov Chain Usage Model and generation method

Re-define the Markov Chain Usage Model to generate Markov Chain Usage Model from De-sequence diagram for testing distributed systems.

Definition12. Let Markov Chain Usage Model as: MC = (S, L, Trans, S_0, F), S is the state, L is the state transition label, Trans is the transition, S_0 is the initial state, and F is the final state.

Definition13. Let state as: $s \in S$ = (Sna, La), Sna is the state name, and La is the state mark.

Definition14. Let state transition label as: $l \in L$ = (C_it_j, Tc, Mna, Pf), C_it_j is the time point, Tc is the time constraint equation, Mna is the massage name, and Pf is the probability of state transition.

Definition15. Let state transition as: $S \times L \rightarrow s$, Trans(s, l) = v, it means state s use l(state transition label) to get to state v.

After that we propose a method to generate Markov Chain Usage Model from De-sequence diagram.

Method2. Choose one of cases we constructed to perform the following operations.

1. Set initial state and final state.

2. Set Pre-condition of transition to be the event of state transition, Post-condition of transition to be arriver state, and enter the state name.
3. Construct the state transition label like l = <$C_i t_j$, Tc, Mna, Pf>.
4. The transition of time violation accordance with the provisions of Tv, and the probability of time violation is been set at 0.1 this time.

Finish other cases like this. At last, we propose a method to combine each Markov Chain Usage Model of cases we constructed.

Method3. First, use the CPre and CPost (defined in Definition5) to describe sequence relationship of cases. If the CPost of case A is the same as the CPre of case B, that case A is the foregoing case of case B, and case B is the immediate successor case of case A. Then, combine them together to build Markov Chain Model for whole system. Take Markov Chain Usage Model initial state of the case to be the final state of its foregoing case, and take the final state of the case to be the initial state of its immediate successor case. If the case does not have the foregoing case or immediate successor case, maintain the status quo.

3. Application Examples

We use E-commerce system and Factory automation system as application examples to confirm the usefulness of the proposed method. We generate the test cases with roulette test case automatic generation method [3] based on two kinds of Markov Chain Usage Model.

3.1 E-commerce System

E-commerce system [4] have three parts distributed in three nodes, it is SS(Supplier Server) in node1, DS(Data Server) in node2, and DC(Data Control) in node3. SS sent commodity query to DS, DS received the message and let DC to query commodity information, DC queried the commodity information and sent it to DS, DS received the information and sent it to SS. From node2 to node3 have three second network delay.

We generate the test cases for E-commerce by building Markov Chain Usage Model from De-sequence diagram as follows:

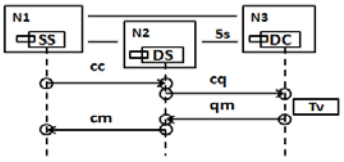

Fig.1:De-sequence diagram of E-commerce system.

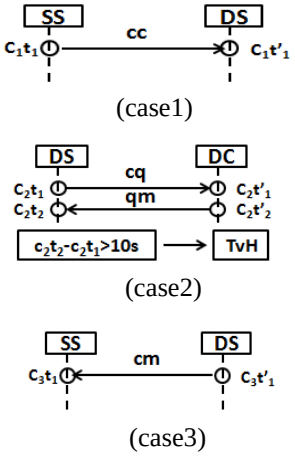

Fig.2:Cases of E-commerce system.

Fig.3:Markov Chain Usage Model of case1.

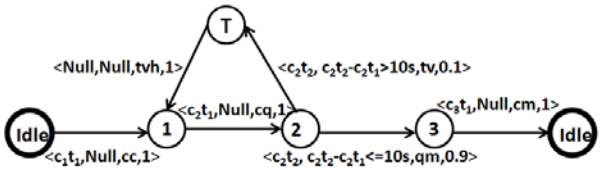

Fig.4:Markov Chain Usage Model of E-commerce system.

1. Construct the De-sequence diagram for the E-commerce system shown in Fig.1 use the Definition1 to Definition11.
2. Divide the system into three cases shown in Fig.2 use the Method1.
3. Choose the first case to generate its Markov Chain Usage Model shown in Fig.3 use the Definition12 to Definition15 and Method2.
4. Generate Markov Chain Usage Model for other cases like the first case.

5. Combine them together to build Markov Chain Usage Model for whole system shown in Fig.4 use the Method3.
6. Generate test cases with roulette method (one time to run it for time violation this time).

Then, we can have two test cases:{cc, cq, tv, tvh, cq, qm, cm} and {cc, cq, qm, cm}. {tv, tvh, cq} in test cases means to test the communication delay. The Markov Chain Usage Model built by the conventional method has only one test case:{cc, cq, qm, cm}.

3.2 Factory Automation System

Factory automation system [4] have five parts distributed in three nodes, SS(Supplier Server) is in node1, PS(Parts Supply) is in node2, WC(Workstation Controller), WC1(Workstation Controller1) and PC(Package Controller) are in node3. Have an order request, C will sent parts query to PS, PS feedback information to C, C is waiting until parts are enough. Then C ask WC whether can work or not and waiting until WC can work. After that, C sent a request to PS for giving parts to WC. When WC finished its work, it ask WC1 whether can work or not and waiting until WC1 can work, then WC sent commodity to WC1. When WC1 finished its work, it ask PC whether can work or not and waiting until PC can work, then sent commodity to PC. From node1 to node2 have two second network delay, from node2 to node3 have three second network delay, and from node1 to node3 have four second network delay.

We use the same method with E-commerce system. Construct De-sequence diagram for the Factory automation system shown in Fig.5, divide the system into eight cases, generate Markov Chain Usage Model for each case, and combine them together to build Markov Chain Usage Model for whole system shown in Fig.6.

Then, we can have four test cases:{cqp, tv, tvh, cqp, qmp, cqwc, tv, tvh, cqwc, qmwc, sr, sd, cqwc1, qmwc1, sdwc, cqpc, qmpc, sdwc1}, {cqp, tv, tvh, cqp, qmp, cqwc, qmwc, sr, sd, cqwc1, qmwc1, sdwc, cqpc, qmpc, sdwc1}, {cqp, qmp, cqwc, tv, tvh, cqwc, qmwc, sr, sd, cqwc1, qmwc1, sdwc, cqpc, qmpc, sdwc1}, and {cqp, qmp, cqwc, qmwc, sr, sd, cqwc1, qmwc1, sdwc, cqpc, qmpc, sdwc1}. The Markov Chain Usage Model built by the conventional method has only one test case:{cqp, qmp, cqwc, qmwc, sr, sd, cqwc1, qmwc1, sdwc, cqpc, qmpc, sdwc1}.

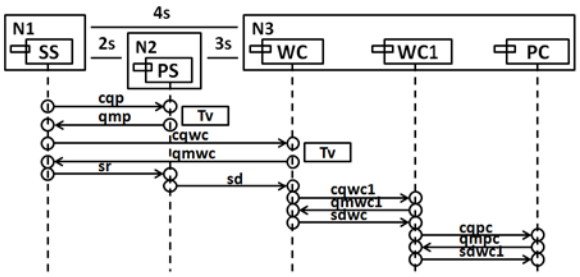

Fig.5:De-sequence diagram of Factory automation system.

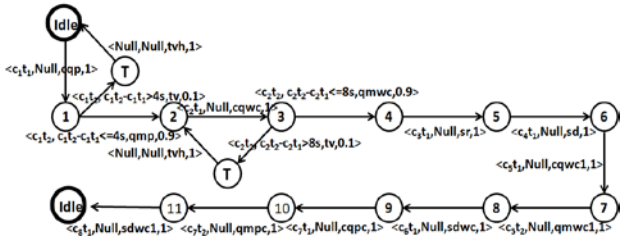

Fig.6:Markov Chain Usage Model of Factory automation system

4. Discussion

T Reference [3] is also proposed a method to build Markov Chain Usage Model from UML diagram, but the method does not discuss the condition of distributed systems. In this paper, we have established the time points to describe the interactive time and sequence between objects and built a new UML diagram to describe the dependencies of distributed systems. We have got the test case included {tv, tvh, cq} in the example of the E-commerce system, and {tv, tvh, cqp}, {tv, tvh, cqwc} in the example of the Factory automation system that the method of reference [2] cannot have. Hence, from comparing results we can know, our proposed method can find the fault of communication delay that the conventional method cannot find. Consequently, we have confirmed that the proposed method can improve the reliability of distributed systems.

5. Conclusion

In this paper, we aim to improve the reliability of distributed systems by testing the communication delay. To achieve this aim, we have proposed a new method to automatically build Markov Chain Usage Model by

establishing the time points and building a new UML diagram for testing communication delay of a distributed system. By adapting this method to practical application examples, we have confirmed that we can test the communication delay of distributed systems that the conventional method cannot test, and that the proposed method can improve the reliability of distributed systems.

Future issues are as follows:

- Get the probability of time violation.
  We did not discuss the probability of time violation, just set it at 0.1 in this paper. We need to propose a method to get the probability of time violation.
- Get the Coverage of time violation
  We did not discuss the coverage of time violation, just run it one time in this paper. We need to think the coverage of time violation to let the test case accurate even more.

References

1. Grady booch, James Rumbaugh, and Ivar Jacobson, The Unified Modeling Language User Guide Second Edition, Addison-Wesley (2005).
2. Wu Caihua, Liu Juntao, Peng Shirui, Li Haihong, Deriving Markov Chain Usage Model from UML Model (in Chinese). Journal of Computer Research and Development (2012), 49(8): 1811-1819.
3. Lei Hang, Chen Limin: Test Case Generation Based on Markov Chain Usage Model (in Chinese). Journal of University of Electronic Science and Technology of China (2011), 40(5): 732-736.
4. Hassan Gomaa, Designing Concurrent, Distributed, and Real-Time Applications with UML, Addison-Wesley (2000), 567-622.