

# Evaluating Reliability of System Sequence Diagram Using Fuzzy Petri Net

<sup>1</sup>H.Nematzadeh, <sup>1</sup>Safaai Bin Deris, <sup>2</sup>H.Maleki, <sup>3</sup>Z.Nematzadeh

<sup>1</sup>Department of Computer Science & Information System, Universiti Teknologi Malaysia,

<sup>2</sup>Science&Research Tehran Branch Islamic Azad University

<sup>3</sup>Allame Mohaddes Noori Institute of Higher Education, Noor, Iran

kamyar61@gmail.com

**Abstract-**Since UML is semi formal, many researches and effort have been performed to transform this language in to formal methods including Petri nets. Thus, the operation of verification and validation of the qualitative parameters could be achieved with more accuracy. Since the majority of the real world information is uncertain, therefore fuzzy UML diagram has been extensively used by system analyzer. This paper attempts to transform system sequence diagram created in fuzzy UML into fuzzy Petri net. Then the reliability is calculated.

**Keywords:** Software engineering, Fuzzy UML, Fuzzy Petri net, system sequence diagram, reliability

## I. INTRODUCTION

UML is famous modeling language, it is used to model systems, it has several diagrams and it is momentous to software engineering[5]. Even though UML is quite useful in software engineering it has its own drawbacks which make UML difficult to be evaluated and verified. This problem is more critical for control, critical, reactive and real time systems. Several researches have been performed to tackle with the semi-formal problem of UML. Some of these researches have only used a transformation algorithm, which transforms the created UML model into a Petri net as a mathematical and formal model that, in turn, contains the visual aspect of modeling and pursues the verification operations with further ability [1-8]. Some of the researches in this field besides representing a transformation algorithm (or without representing an algorithm and only by using the available Algorithm); evaluate the capability of the nonoperational parameters and commonly

qualitative parameters on the obtained Petri nets of the UML model created [9-12]. In [23] fuzzy petri net is used for intelligent database. In our previous researches [14-17] besides of studying and presenting transformational patterns for some kinds of usual UML diagrams, especially state diagrams and activity diagrams[1], we presented methods for evaluating some qualitative parameters. In this paper, due to the growing process of using UML diagrams in fuzzy model, we centralized on this kind of diagrams and with the significant ability of Petri nets in semi-formal UML model formalization we present a pattern to transform fuzzy system sequence diagram to fuzzy Petri nets. This work is a sequel to our previous attempt to convert activity diagram in fuzzy UML to fuzzy Petri net[1]. In a proposed general pattern [18, 19]. Since the real world information is mostly uncertain, in many case these type of information cannot be modeled by UML. Recently, a model named fuzzy Model has been introduced [20-22] which has the UML characteristics, is also able to model uncertain concepts.

## II. FUZZY SYSTEM SEQUENCE DIAGRAM

A system sequence diagram is a fast and easily created artifact that illustrates input and output events related to the systems under discussion. System sequence diagrams [24] like use cases and system contracts describe *what* a system does, without explaining how it does it. A simple system sequence diagram comprises: actor, system, and messages. Each message itself has events and conditions. This diagram uses fuzzy rules for transforming the state of an object to another state. A fuzzy rule can be expressed as in (1). Based on (1) we defined FSSD in fig 1.

Rule = If<condition list>Then<event list> (1)

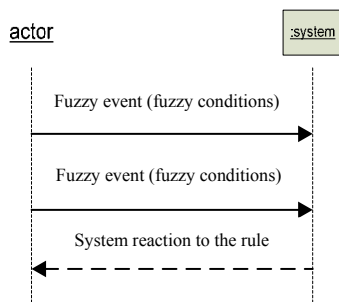


Fig. 1. FSSD based on (1)

### III. FUZZY PETRI NET

We introduce the following fuzzy Petri net (FPN) structure to model fuzzy rules [23] :  $(P, P_s, P_e, T, T_F, TRTF, A, I, O, TT, TTF, AEF, PR, PPM, TV)$ , where

(I)  $P$  is a finite set of fuzzy places. Each place has a property associated with it, in which

- $p_s \subset p$  is a finite set of input places for primitive events.
- $p_e \subset p$  is a finite set of output places for actions or conclusions.

(II)  $T$  is a finite set of fuzzy transitions. They use the values provided by input places and produce values for output places.

(III)  $TF$  is a finite set of transition functions, which perform activities of fuzzy inference.

(IV)  $TRTF : T \rightarrow TF$  is transition type function, mapping each transition  $\in T$  to a transition function  $\in TF$ .

(V)  $A \subseteq (P \times T \cup T \times P)$  Is a finite set of arcs for connections between places and transitions. Connections Between the input places and transitions ( $P \times T$ ) and connections between the transitions and output places ( $T \times P$ ) are provided by arcs. In that:

- $I : P \rightarrow T$  is an input mapping.
- $O : T \rightarrow P$  is an output mapping.

(VI)  $TT$  is a finite set of fuzzy token (color) types. Each token has a linguistic value (i.e., low, medium and high), which is defined with a membership function.

(VII)  $O : T \rightarrow P$  Is token type function, mapping each fuzzy place  $\in P$  to a fuzzy token type  $\in TT$ . A token in a place is characterized by the property of the place and a level to which it possesses that property.

(VIII)  $AEF : Arc \rightarrow$  Expression is arc expression function mapping each arc to an expression, which carries the information (token values).

(IX)  $PR$  is a finite set of propositions, corresponding to either events or conditions or actions/conclusions.

(X)  $PPM : P \rightarrow [0,1]$ , is a fuzzy place to proposition mapping, where  $|PR| = |P|$ .

(XI)  $TV : P \rightarrow [0,1]$  is truth values of tokens ( $\mu_i$ ) assigned to places. It holds the degree of membership of a token to a particular place.

### IV. TRANSFORMATION ALGORITHM

**Step1.** First for each message in this diagram, its event and conditions must be found. The events and conditions calculated for the dream activity is represented in Table1.

**Step2.** First we need to check the correctness of the conditions, therefore we should provide a mapping to do that. For each condition we provide a transition that is responsible to check the correctness of it and the result will go to another place. That means the token with the given fuzzy amount continues its lifecycle. And the token may have the value from 0 to 1, that means:  $0 \leq \text{token value} \leq 1$ . The result at the end of analyzing condition  $C$  will be a fuzzy amount based on the condition, Sometimes we have more than one condition, so depends on fuzzy logical operations we choose the appropriate function. The concept is in equations in (2):

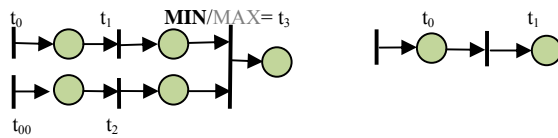
$$\begin{aligned} \text{OR} &= \mu A \cup B(x) = \max [\mu A(x), \mu B(x)] \\ \text{AND} &= \mu A \cap B(x) = \min [\mu A(x), \mu B(x)] \end{aligned} \quad (2)$$

In fig 2, two different common cases are depicted. Likewise we can play with the rules and create more complicated and complex logical operation by extending the figure based on the number of rules and order of logical operation.

**Step3.** The next step is to run the event/events if the conditions are met. Here the antecedent fuzzy amount is used to evaluate the consequent. The transition here is used to apply the antecedent truth value to the consequent membership function. Aggregation occurs here as in fig 3. The defuzzification process will be done here because the final output of a fuzzy system should be a crisp number. The defuzzification process is done based on center of gravity test. Center of gravity can be expressed as equation (3).

$$\text{COG} = \frac{\int_a^b \mu(A) \times dx}{\int_a^b \mu(A) dx} \quad (3)$$

TABLE I  
EVENT AND CONDITION FOR DREAM ACTIVITY



a. If (C is C<sub>1</sub>) AND/ OR then...      b. If C is C<sub>1</sub> then ...

Fig. 2. Modeling antecedent of the fuzzy rule (a) without and (b) with logical operation

If C is C<sub>1</sub> then E is E<sub>2</sub>  
If C is C<sub>2</sub> then E is E<sub>2</sub>

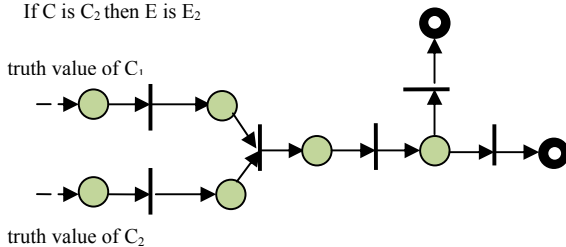


Fig. 3. Modeling consequent of fuzzy rule with aggregation

Here we use fuzzy petri net to calculate the data flow. As brief description of fig 3 we can say t1 uses C<sub>1</sub> to adjust and clip consequent membership function of E<sub>1</sub>, on the other hand, it calculates the consequent of rule 1. t2 does same but for E<sub>2</sub>. t3 aggregates the two rules. t4 does defuzzification. t5 and t6 both are responsible to check to see if the correct condition that needs to trigger the system occurs or not. If it doesn't occur t5 passes the token to its end place and if it occurs t6 passes it to the system. The result of this step (the amount in the output place of t6 ) which is a strength fuzzy amount will be exerted from actor to the system as a black box or from system to the actor as a result of a message

## V. RELIABILITY USING FUZZY PETRI NET

For calculating reliability of the system we may define a success rate,  $f$ , for the transition in CPN. Success rate specifies the probability of firing transition if something wrong does not happen, on the other hand  $1-f$  is the failure rate, the probability of data loss. The token in the input of the transition  $t$  is assumed that carries the amount which stands for the accumulation of the success rate up to that place. When transition fires the amounts which represents

the success rate will be changed to  $D*f$ , i.e. the token has got a new

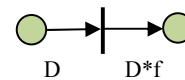
Rule	Event	Condition
R1	E is E <sub>1</sub>	C <sub>1</sub>

probability of firing;  $D*f$  instead of  $D$ . Fig 4. Shows the concept.

## VI. CASE STUDY

### A. Calculations

The following case study is chosen because it is familiar, but rich architectural problems, and thus allows one to concentrate on how to do analysis, rather than explain the problem and domain. A CRS system is a computerized application used to record rents and handle payments; it is typically used in web based car renting systems. It includes hardware components such as a computer and software to run the system. It interfaces to various service



D: accumulation of success rate = Reliability  
Fig 4. Reliability using fuzzy petri net

applications. These systems must be relatively fault-tolerant; that is, even if remote services are temporarily unavailable, they must still be capable of capturing data and handling at least cash payments.. After studying the use cases and use case diagram, the main successful scenario is chosen by the analyzer for SSD. Analyzer chose the SSD in fig 5.

Now gradually we are going to transform each sequence into fuzzy message. The customer should be able to initiate a new rent if two requirements are provided. The event and conditions of the first message are derived. Like the first actor-system message, other actor-system messages and the fuzzy rules are created by fuzzy specialist. For system-actor messages we don't need to do the same mapping because in SSD, system is black box, so this is just the result that system produces is important and we don't need to know how it is created. This message is based on the result from the previous step. System reaction to the rule is done via a simple message but we should define two important things; first value(s) that should be returned from system to actor is associated with the previous message and in case of no proper result from the previous message, customer should resend the previous message. Fuzzy

rules for three messages are in fig 6,7 and 8 respectively.

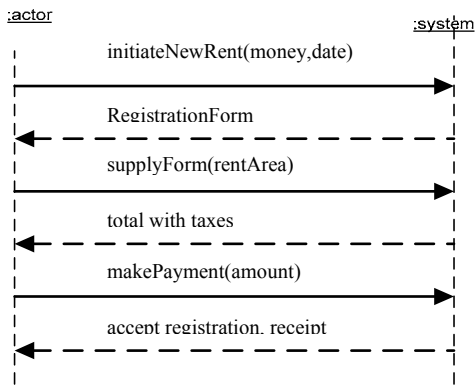


Fig 5. SSD for car renting system

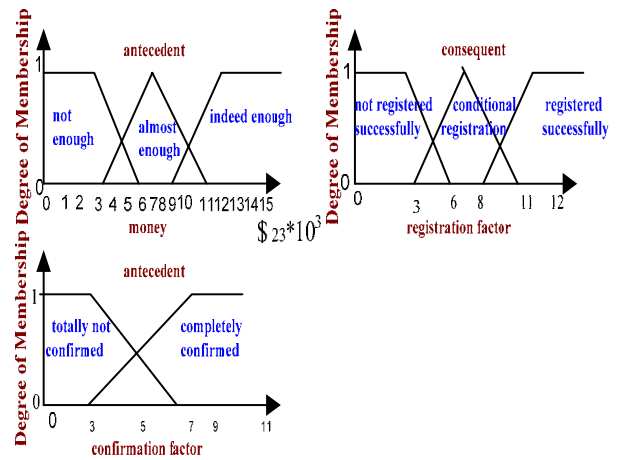


Fig. 8. Fuzzy rule for 3<sup>rd</sup> actor-system message

And fig. 9. shows the Fuzzy Petri net for SSD in fuzzy UML.

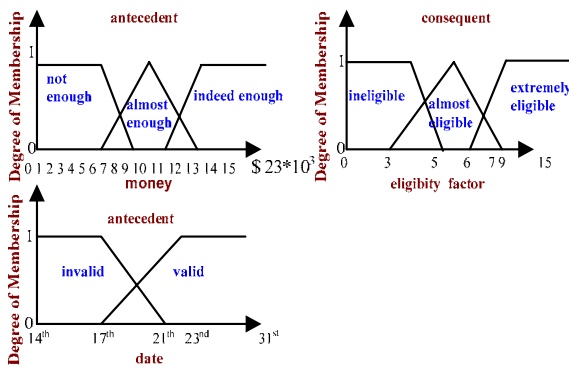


Fig. 6. Fuzzy rule for 1<sup>st</sup> actor-system message

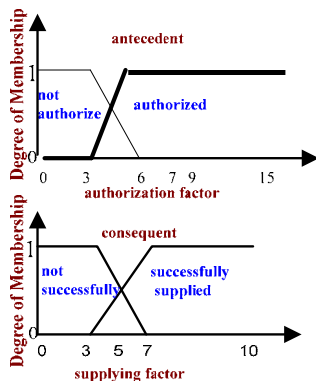


Fig. 7. Fuzzy rule for 2<sup>nd</sup> actor-system message

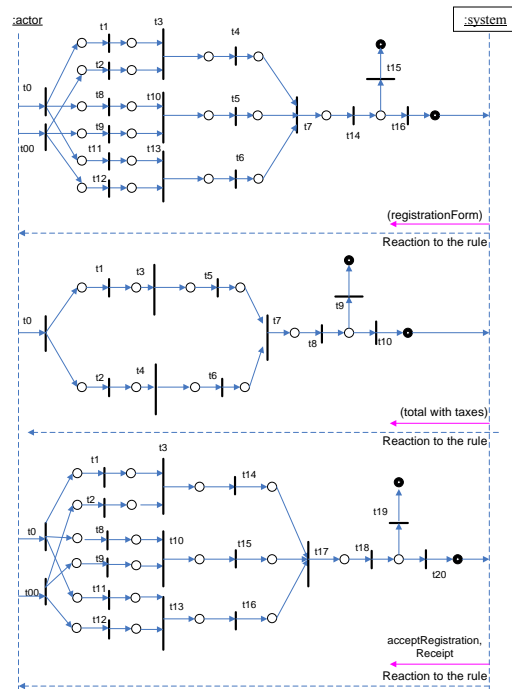


Fig. 9. Fuzzy Petri net for SSD in fuzzy UML

Fig 9. Shows the final fuzzy system sequence diagram modeled with fuzzy petri net. To calculate the reliability of each incoming message or actor-system message, we can follow the formula given in V. With the input money s \$7000 and the date 18<sup>th</sup> , we may have the following fuzzy values:

$$\begin{aligned} \mu(\text{money}=\text{indeed enough}) &= 0 & \mu(\text{date}=\text{valid}) &= 0.19 \\ \mu(\text{money}=\text{almost enough}) &= 0.3 & \mu(\text{date}=\text{invalid}) &= 0.64 \end{aligned}$$

$\mu_{(\text{money}=\text{not enough})} = 0.55$

We assume all the transitions have the probability of firing 0.95 except t0 and t00 with the reliability of 1.

path: The tokens may pass all the transitions except t15 and they may pass the rest once. One important point here is that in case of many inputs and one output as in t3 we assume D in fig. 4 the minimum of the input reliabilities. This is because t3 should have its both inputs if it wants to fire. Therefore the reliability of a token after passing each transition for message one in the fuzzy SSD for the given inputs is as follow: t0 =1, t00 =1, t1 = 0.95, t2 = 0.95, t8 = 0.95, t9 = 0.95, t11=0.95, t12 = 0.95, t3 =0.90 , t10 =0.90 , t13 = 0.90, t4 = 0.85, t5 = 0.85, t6 = 0.85, t7 = 0.81, t14 = 0.77, t16 = 0.73

The final reliability of message one with mentioned input is 0.73 because it is the last transition, token passes. For each message we can calculate the reliability based on the path a token may move and success rate of the transitions token encounters in its lifecycle. And the reliability of SSD is the reliability of its messages individually.

#### B. Discussions on the Result

The final reliability of the 1<sup>st</sup> message is 0.73 which is very low, that is because the reliability of each transition is 0.95. The final reliability of SSD depends on the success rate of each transition in each incoming message. The developer of the system should create more reliable components in other word transitions with better success rate. In order to enhance the reliability more exact software classes should be programmed that is usually done in the implementation phase. In RUP these kinds of shortcomings will be solved during different iterations. Reliability plays an important role in purchasing software and it should be considered by system Analysts. This is very significant while dealing with crucial controlling systems. Analysts of the software can benefit from this research.

### VII. CONCLUSION AND FUTURE WORK

#### A. Conclusion

The study concludes to model and evaluate the UML system sequence diagram. System sequence diagram first modeled to fuzzy petri net and then the model was analyzed based on reliability. The previous study have shown us that we cannot derive nonfunctional parameters from a functional diagram and this limited developers to find out the quality values of SSD. But through the formalism of the Petri net, analysis of

SSDs in terms of nonfunctional parameters is done. At the end the result of these experiments were assessed.

#### B. Future work

As a future work, other UML diagrams specially the behavioral diagrams of UML and other nonfunctional parameters or can be evaluated through petri net. Besides UML diagrams, software architectures will be studied later through the same technique.

#### REFERENCES

- [1] H. Motameni, A. Movaghar, I. Daneshfar and H. Nemat Zadeh, " Mapping to Convert Activity Diagram in Fuzzy UML to Fuzzy Petri Net," World Applied Sciences Journal 3 (3): 514-521, 2008 ISSN 1818-4952 © IDOSI Publications, 2008.
- [2] Faul M. B, "Verifiable Modeling Techniques Using a Colored Petri Net Graphical Language," Technology Review Journal, spring/summer, 2004.
- [3] M. Shin, A. Levis, and L. Wagenhals, "Transformation of UML-Based System Model into CPN Model for Validating System Behavior," In: Proc. of Compositional Verification of UML Models, Workshop of the UML'03 Conference, California, USA, Oct. 21, 2003, in press.
- [4] S. Bernardi, S. Donatelli, and J. Merseguer, "From UML Sequence Diagrams and Statecharts to Analysable Petri Net Models," ACM Proc. Int'l Workshop Software and Performance, pp. 35-45, 2002.
- [5] R. Eshuis, "Semantics and Verification of UML Activity Diagrams for Workflow Modelling," Ph.D. Thesis, University of Twente 2000, in press.
- [6] R. G. Pettit and H. Gomma, "Validation of dynamic behavior in H.UML using colored Petri nets' UML," (2000), Zaragoza, Spain, , 2000, 295-302.
- [7] J. Saldhana, and S. M. Shatz, "UML Diagrams to Object Petri Net Models: An Approach for Modeling and Analysis," Proc. of the Int. Conf. on Software Eng. And Knowledge Eng. (SEKE), Chicago10 – 103, 2000, in press.
- [8] M. Elkoutbi and K. Keller Rodulf, "Modeling Interactive Systems with Hierarchical Colored Petri Nets," 1998 Advanced Simulation Technologies Conf., Boston, MA, 1998 432- 437 .
- [9] L. Bernardinello and F. De Cindio, "A Survey of Basic Net Models and Modular Net Classes," LNCS, vol. 609, Springer-Verlag, 1992, p.609.
- [10] S. Balsamo et al, "Model-Based Performance Prediction in Software Development: A Survey"IEEE Trans.Magn., VOL. 30, NO. 5, May, 2004, p295.
- [11] J. Merseguer, , J. P. LopezGrao and J. Campos, "From UML Activity Diagrams To Stochastic Petri Nets: Application To Software Performance Engineering," ACM, WOSP 04 January 1416, 2004.
- [12] K. Fukuzawa, et al, "Evaluating Software Architecture by Colored Petri Net," Dept. of Computer Science, Tokyo Institute of Technology Ookayama 2-12-1, Meguro-uk, Tokyo 152- 8552, Japan 2002.
- [13] J. Merseguer, S. Bernardi, J. Campos and S. Donatelli, "A Compositional Semantics for UML State Machines Aimed at Performance Evaluation," M. Silva, A. Giua and J. M Colom (eds.), Proc. of the 6th Int. Workshop on Discrete Event Systems(WODES'02), Zaragoza, Spain 2002, 295-302.
- [14] H. Motameni et al, "Mapping State Diagram to Petri net: , "An Approach Tousemarkov Theory For Analyzingnon-Functional

- Parameters,"IEEE,internationalconferenceon, computer information and system science,december 4\_14 2006 , university of bridgport, USA, in press.
- [15] H. Motameni et al, "Using Markov Theory For Deriving Non-Functional Parameters On Transformed Petri Net From Activity Diagram, ",proc of software engineering conference Russia, 16-17 November 2006, in press.
- [16] H. Motameni, M. Zandakbari and Movaghar,"Deriving Performance Chain, " ICCSA 2006 Proceeding of 4<sup>th</sup> International Conference On Computer Science and Its applications, San Ddiego,California , 2006,in press.
- [17] H. Motameni, "Evaluating UML State Diagrams Using Colored Petri Net" SYNASC'05, 2005, in press.
- [18] Object Management Group "UMLTM Profile for Schedulability, Performance, and Time Specification ,"OMG Document, Version 1.1, January 2005.
- [19]J. Rumbuaugh, M.Blaha and W.Premarlani,"Object-Oreinted Modeling And Design, " prentice hall , Englewood Cliffs,nj,USA, 1991,in press.
- [20] I. Wang, "Fuzzy UML,"Seminararbeit,Sommersemester 2005, in press.
- [21] Zongmin Ma. "Fuzzy Information Modeling With the Uml". Idea, 2005, in press.
- [22] Z. Ma, "Extending UML For Fuzzy Information Modeling In Object\_Oriented Database, " theories and practices,idea group publishing, 2004, in press.
- [23] B. Korpeoglu and A. Yazici, "A Fuzzy Petri Net Model For Intelligent Database," Data & Knowledge Engineering, Elsevier, 2006.
- [24] C.larman "applying UML and patterns" an introduction to object oriented Analysis and design and the unified process, second edition, 1998.