World Applied Sciences Journal 3 (5): 797-805, 2008
ISSN 1818-4952
© IDOSI Publications, 2008

A Novel Method for Behavior Modeling in Uncertain Information Systems

[1]Ali Haroonabadi, [2]Mohammad Teshnehlab and [3]Ali Movaghar

[1] Azad University, Science and Research Branch, Tehran, Iran
[2]Department of Electrical Engineering, Khaje Nasir Toosi University, Tehran, Iran
[3]Department of Computer Engineering, Sharif University of Technology, Tehran, Iran

Abstract: None of the processing models in the software development has explained the software systems performance evaluation and modeling; likewise, there exist uncertainty in the information systems because of the natural essence of requirements and this may cause other challenges in the processing of software development. By definition an extended version of UML (Fuzzy-UML), the functional requirements of the software defined uncertainly would be supported. In this study, the behavioral description of uncertain information systems by the aid of fuzzy-state diagram is crucial; moreover, the introduction of behavioral diagrams role in F-UML is investigated in software performance modeling process. To get the aim, a fuzzy sub-profile is used.

Key words: Fuzzy sub-profile . fuzzy system . software development model . software performance evaluation . UML

INTRODUCTION

Nowadays, the extent of Information systems in the different usages is undeniable. Due to systems development UML, Unified Modeling Language, that support the object oriented concepts, proposed by Rumbaugh, Booch and Jacobson in the mid of 1990 decade. The unification of symbols, techniques and propounded instructions in UML defined it as a general language for different usages in object oriented approach. Although UML has been extended a lot nowadays, it has two important shortcomings:

The applied techniques in UML are useful for certain problems. In other respect, uncertainty is considered in many information systems. Since information systems resolve the users requirements and uncertainty is considered cause of user requirements natural essence, if we include the uncertainty in UML, it increase the rate of effectiveness.

Since UML is not a formal model, software systems evaluation is not possible directly. Although OMG that is responsible for UML development, introduced a profile to support performance concepts in 2002 [1], it needs to transform pragmatic model to formal model for performance evaluation purposes.

The proposed idea is considered in Fig. 1 (to overcome the mentioned problems). The attention of the paper is on the gray parts in Fig. 1.

Behavior modeling in information systems is very important. In system behavior modeling, fuzzy state diagram is the most important part among the other parts. This diagram is able to support system

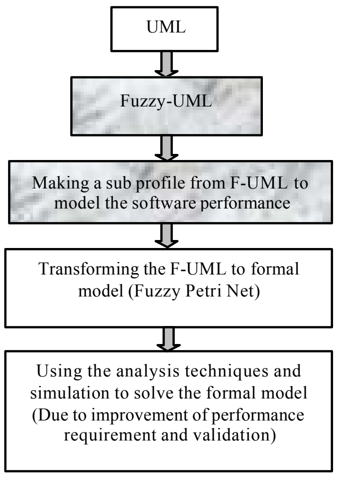
UML
Fuzzy-UML
Making a sub profile from F-UML to model the software performance
Transforming the F-UML to formal model (Fuzzy Petri Net)
Using the analysis techniques and simulation to solve the formal model (Due to improvement of performance requirement and validation)

Fig. 1: The presented idea in this paper

performance requirements alone. In "incorporating imprecise reasoning into state diagram" part, a fuzzy model from the diagram is presented and in "modeling fuzzy states in an industrial greenhouse" part, an inspection is done in case study form.

In the selected approach, UML diagrams not only support the uncertain functional requirement but also specify the software performance requirements. The *Software Performance Engineering (SPE)* community introduced UML as a reference to of describe the software systems in its international workshop [2]. In [3] SPE is defined as a method (a quantitative and systematic approach) to produce the software systems in a way to observe the qualitative objectives. In this

Corresponding Author: Ali Haroonabadi, Azad University, Science and Research Branch, Tehran, Iran

paper, the SPE approach that lots of activities have been done on it is reviewed. In addition to introducing F-SPE (Fuzzy-SPE) method, what is important in this paper, is specifying the role of some F-UML diagrams (use case, sequence) in performance from the object oriented view.

In the second section of the paper, we talk about entering the uncertainty in UML and review of the fuzzy subjects from two aspects of information systems: structural and behavioral. To consider the additional subjects for this section [4-7]. The third section introduces a sub profile from F-UML to model the software performance, that it helps transform the F-UML to the uncertain formal model (fuzzy Petri Net) and the performance requirements and system functionality are improved after solving the formal model.

PRESENTATION OF A FUZZY MODEL TO DEVELOP THE UNCERTAIN SYSTEMS

UML supports both behavioral and structural aspects of system; therefore, fuzzy concepts are propounded in two sections: fuzzy structure and fuzzy behavior. To support system fuzzy structure, a fuzzy data model and to support system fuzzy behavior, the models are needed to support system functionality fuzzily.

The first part of this section is about data modeling, that the uncertainty in data structure is entered by presenting a class diagram fuzzily. The fuzzy behavior modeling is inspected in the second part. This subject is realized by presenting the use case, sequence and state diagrams fuzzily. The other diagrams have been investigated in [4, 5].

F-UML data model: The class diagrams in UML are the logical models. They describe the system main structure. The classes and the relationships among them consist of the elements in class diagram. By entering the uncertainty into these elements, the F-UML data model is produced. Considering Zongmin's finding [7] in the context of classes, the three levels of fuzziness are defined as follows:

Fuzziness in the extent to which the class belongs in the data model as well as fuzziness on the content (in term of attributes) of the class.

Fuzziness related to whether some instances are instances of a class; even though the structure of a class is crisp, it is possible that an instance of the class belongs to the class with degree of membership.

The third level of fuzziness is on attribute values of the instances of the class; an attribute in a class defines a value domain and when this domain is a fuzzy subset or a set of fuzzy subset, the fuzziness of an attribute value appears.

The attribute or the class name in the first level should be described by this phrase

WITH mem DEGREE

Where, $0 \leq mem \leq 1$. This value shows the degree of belonging the attribute to the class or the class to the data model.

The second level of fuzziness, the membership degree in an instance of the class that it belongs to the class should be specified. So an additional attribute in the class is defined for representation of the instance membership degree to the class which its domain is [0,1]. This special attribute has specified with μ. The classes with the second level of fuzziness have specified by a rectangle that its lines are dash.

In the third level, a *fuzzy* keyword is appeared in front of the attribute.

Figure 2 shows the *banking account* fuzzy class. In the mentioned class, the *credit* attribute could have the fuzzy value (the third level of fuzziness). In the other hand, the credit attribute is a linguistic variable and it has a domain like fuzzy sets (for example: little/much).

The account *type* specifies the membership degree of *credit* attribute to the class (the first level of fuzziness):

"*Credit With 0.8 membership Degree*"

Finally, μ attribute specifies the membership degree of a class instance to the class (the second level of fuzziness).

The relations among the classes are divided into four categories and they are propounded fuzzily [5, 7]: fuzzy generalization, fuzzy association, fuzzy aggregation and fuzzy dependency.

Presentation of a fuzzy model from the system behavior: There are different approaches to describe the system behavior. The system behavior can be illustrated in the information systems by the used techniques in UML. The steps that UML presents for certain systems development are briefly extended to the uncertain systems and the F-UML version is improved afterward.

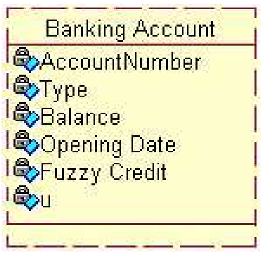

Fig. 2: A fuzzy class of banking account

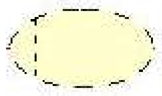

Fig. 3: The fuzzy use case symbol in F-UML

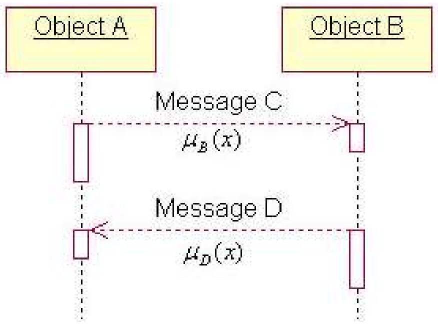

Fig. 4: The fuzzy messages in F-UML

Specifying the uncertain requirements of the system: To assign the system behavior, its functionality requirements are specified at first. The requirements extraction in object orientation is service oriented. When a service is presented uncertainly, the fuzzy use case will be propounded. The showed symbol in Fig. 3 is used to describe this concept in F-UML.

After specifying the certain and uncertain use cases, we perform to draw the fuzzy use case diagram [5].

Assigning the flow of events to realize the fuzzy use cases: A use case is a sequence of actions that a system does to prepare an observable result for user [8]. In the other hand, for each use case, the scenarios could be set that explain the steps of the use case. In UML, sequence diagram is used to realize the use cases. If the selected use case is uncertain, the sequence diagram will be uncertain too. The dash lines for uncertain messages representation in mentioned diagram are used. Figure 4 represents this subject.

The messages in sequence diagram explain the methods. The uncertainty in method has two level of fuzziness:

The first one is that the method belongs to the object with membership degree and the second one is that the method essence is fuzzily. In Fig. 4, the method *C* which is produced from message *C* belongs to the object B with the membership degree between 0 and 1 (first level fuzziness).

In other respect, the essence of D message is also uncertain (second level fuzziness). For example, “approval of credit account” message (method) has fuzzy domain and is uncertain in a bank system to receive a loan.

The use case membership degree is equal to t-norm messages membership degrees that exist in sequence diagram. For example, the use case membership degree that is shown by sequence diagram of Fig. 4 is equal to:

$$t[\mu_B(x), \mu_D(x)] = \mu_{B\cap D}(x)$$

Where, $t:[0,1]\times[0,1]\rightarrow[0,1]$

In addition to above-mentioned cases, it is needed to make fuzzy decisions in many software systems. The propounded fuzzy scenarios in F-UML present a solution for this subject. With specifying the effective parameters in a scenario, the scenarios are transformed to the compositional fuzzy rules. An example about this subject was mentioned in the fourth section of the paper.

Incorporating imprecise reasoning into state diagram: The state diagram is one of the UML diagrams used to model system behavior.
State diagram consist of four main elements:

- States which define behavior and may produce actions
- State transitions which are movement from one state to another
- Rules or conditions which must be met to allow a state transition
- Input events which may possibly trigger rules and lead to state transition

The performance requirements gathered by modeling the state machines for the systems are sufficient enough to obtain a performance model [9]. Considering the importance of the diagram, imprecise reasoning enters to it. Fuzziness in state diagram is taken in to consideration from three aspects:

Fuzzy state: A fuzzy state is a state with possible degrees of membership $0\leq\mu\leq1$, as opposed to a (crisp) state with an implicit degree of membership of zero or one. That is, a fuzzy state diagram allows the system to be partially in the current state.

Fuzzy condition: The following rules base are supposed:

$$\overset{(1)}{R}u: IF\ x_1\ is\ A_1^l\ and\ x_2\ is\ A_2^l\ Then\ y\ is\ B^l \qquad (1)$$

Where, A^l and B^l are respectively fuzzy sets in $U_i\subset R$ and $V\subset R$, $x = (x_1, x_2)^T\in U$ and y are respectively input and output linguistic variables in fuzzy system.

M is the number of existence rules in fuzzy rule base: l = 1, 2,.., M The degree of truth for a rule's premise is sometimes referred as its alpha. It is computed as follows:

$$\alpha_i = \mu_{A_1^l} \text{and} \mu_{A_2^l}(x_1, x_2) = \min(\mu_{A_1^l}(x_1), \mu_{A_2^l}(x_2))$$

If a rule's premise has nonzero degree of truth (i.e. when the input matches partially the premise of the rule) then the rule is fired.

To increase restriction on conditional part, the threshold value could be defined. If the truth rate of condition become more than threshold value, then transition will be done.

Fuzzy event: A fuzzy event is defined with membership degree between 0 and 1; in other word, a partial of an event comes true.

Condition-Action model: Each fuzzy rule (condition in state diagram) could be led to occur an action. Suppose that:

if FP1 then FA1
if FP2 then FA2
...
if FPn then FAn

Where,
FPi: A fuzzy predicate,
FAi: A fuzzy action proposition represented like "Y is yi"

Where, Y is a fuzzy action linguistic variable and T(Y)={y1,y2,…,yn} is the term set of Y.

There also exists a set of concrete actions A={a1,a2,…,an}. Each ai (i=1,2,…,n) may be implemented, in a real system, as a distinct procedure. All the membership functions of Y are defined over the same arbitrary domain.

Then a map could be created between fuzzy action and concrete action. For example, in Fig. 5(a), fuzzy actions are mapped into related concrete actions.

Execution model: To implement the propounded idea, fuzzification is done after occurrence of the event. For this goal, the linguistic variables in the antecedent part of the rules are evaluated, i.e., the corresponding source data are fuzzified. Then by the use of max-min inference method y* output is formed. To choose and execute the considered action the conclusion is defuzzified and the crisp value is taken (e.g. *r*). In continuation select the fuzzy action that membership function produce highest value:

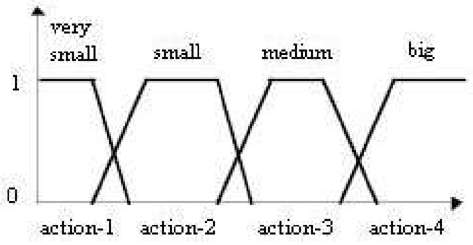

Fig. 5 (a): Fuzzy action and concrete action

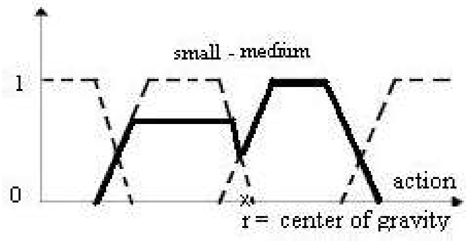

Fig. 5 (b): Fuzzy result action

$$\mu_{y_i}(r) = \max(\mu_{y_1}(r), \mu_{y_2}(r), ..., \mu_{y_n}(r))$$

Finally, the related concrete action for execution is selected. For example in Fig. 5(b), *small-medium* output fuzzy set is considered. By defuzzification (center of gravity method) of the set, crisp value of *r* will be gotten. This value is mapped to *medium* fuzzy action and the related concrete action, action3, is chosen.

F-SPE METHOD

The complete software systems are introduced as the most complex products that were produced by human civilizations [10]. These systems need the formal models to model and analyze.

Although the several process models are proposed in software engineering field, but none of them has propounded the modeling and performance evaluation in software systems at software development life-cycle. It's necessary to say that separate inspection of these two subjects is false [9] without considering to the relationship between them. The propounded profile by OMG was a start point to remove the gap between software evaluation techniques and software development processes. The proposed approach in this paper is a step to connect these two subjects and it can be used with all of the software development process models.

The role of UML diagrams in system performance computations is considered to receive the F-SPE method. UML diagrams are divided into three groups: behavioral diagrams, structural diagrams and

Table 1: A summary of annotation

	Annotation	Kind	Referenced value
Use case diagram	Probability that an actor executes a use case	System usage	Association link
Sequence diagram	Probability of success of a message	Routing rate	Message
	Message size	System load	Message

implementation diagrams. All of the above diagrams relate to the system performance aspects. For describing a system performance completely, behavioral and structural diagrams are considered in the modeling level and the implementation diagrams are propounded when the system hardware configuration is considered [9]. Considering the systems performance studies focus is on the system dynamic aspect, so among the above diagrams, behavioral diagrams have the most prominent role in describing the performance of a system. Table 1 represents a summary of annotation for certain information [11]. This paper presents annotation to uncertain information for two frequently used behavioral diagrams among five behavioral diagrams (use case, sequence, collaboration, activity, state). To inspect the other diagrams, refer to [5].

It should be considered that the performance computations that are proposed fuzzily, are propounded for both type of information systems extended with UML and F-UML (being fuzzy the computation of performance is independent from the system development type).

Supporting performance concepts by an extended version from F-UML is introduced. This version uses a term named *tag definition* that has the following three types: *System load*, *routing rate* and *system usage*

Tagged value permits required information to be able to attach the model elements according to the tag definition format. In the other hand, tagged value prepares this possibility to set annotation related to the performance, on the model elements. In certain information system, each concrete performance annotation in a UML diagram will be specified as a tagged value with the form {a concrete annotation}, that is a quantitative annotation inside braces, for example {20 KB}.

If tagged value is certain, it will include data value from crisp type; otherwise, it will include data value from fuzzy type (the data value domain is fuzzy set). As an example, the annotation named *message size* that is from system load type has *{100k}* data value in certain information; though, it has *little* data value in uncertain information.

Use case diagram: In this diagram, actors, use cases and the relationships between them are inspected. The relationship between the actor and the use case is association, the relationship between the actors is generalization and the relationship among the use cases are generalization, include and extend. Among the propounded use cases in a diagram, some of them are important from performance aspect and they are inspected.

The role of use case diagram in performance evaluation process: By use cases diagram it can be specified that how much each actor uses the system. In [12] a probability is allocated (the probability of executing the use cases by the actor) to each relationship between use case and actor. Suppose that there is a use case diagram with m users and n use cases:

$p_i(i = 1,..,m)$ the frequency of *i*-th user usage from the system

$p_{ij}(j = 1,..,n)$ the probability of using the *i*-th user from *j*-th use case

$$\sum_{j=1}^{n} P_{ij} = 1 \text{ and } \sum_{i=1}^{m} p_i = 1$$

Then the probability of executing the sequence diagram the *x*-th use case is equal to:

$$P(x) = \sum_{i=1}^{m} p_i P_{ix} \quad (2)$$

Toward a fuzzy approach for computations related to performance in use case diagram: Linguistic variables and fuzzy rules for entering uncertainty into the performance computations are used. Considering rules base (1), the selected fuzzy system has *product inference engine*, *singleton* fuzzifier and *center average* defuzzifier:

$$f(x) = \frac{\sum_{l=1}^{M} \bar{y}\left(\prod_{i=1}^{n} \mu_{A_i^l}(x_i)\right)}{\sum_{l=1}^{M}\left(\prod_{i=1}^{n} \mu_{A_i^l}(x_i)\right)} \quad (3)$$

Where, B^l fuzzy set is a natural set with y^{-1} center.
The following fuzzy variables are supposed:

x_1: The *i*-th user usage frequency from system

x_2: The probability of the *i*-th user usage from selected use case y: The probability of using the *i*-th user from the selected use case

Then (for example) the following fuzzy rule can be propounded:
If x_1 is a little high and x_2 is high then y is high.

Defuzzifier is a mapping of B^l fuzzy set in $V \subset R$ (fuzzy inference engine output) to a $y^* \in V$ certain point. This certain point, is the same probability of executing the use case and use it by the i-th actor. The Fig. 6 shows use case diagram with performance annotation (fuzzy approach). Suppose that two actors use the *use case*, in the same way, the probability will be computed for the second actor too. The sum of these two values is the probability of using the selected use case in the system. As it was said in the Table 1, *system usage* is counted as one of the system performance requirements.

In the Fig. 6, y_1 and y_2 are the outputs of fuzzy system and p_1, p_2, p_3 and p_4 are the fuzzy sets.

Sequence diagram: The introductory descriptions about this diagram have described in "assigning the flow of events to realize the fuzzy use cases" part. The sequence diagram has two dimensions. The vertical dimension of diagram shows the time and the horizontal dimension of it shows the class instances (objects).

The role of sequence diagram in performance evaluation process: In this diagram, objects can be put on the one system or different systems. In the centralized system, the spent time to dispatch a message is not important for performance modeling. Although actions need some times to respond the message for computations, investigation to this subject can be considered in state diagram [13]. For distributed systems, the messages traveled through the net, allocate a noticeable time to themselves that can be propounded in system load.

In other respect, a condition could be attached to each message in diagram and based on it the probability of sending message is specified. From the performance view point, this subject leads to attach the routing rate on the message. For example, the messages m2 and m3 are showed in Fig. 7.

Toward a fuzzy approach for computations related to performance in sequence diagram: The response time in distributed systems depends on message size and net speed. To be aware of the message size and net speed as linguistic variables and fuzzy sets (like: large, medium and small) as linguistic variables domain, for each message, the message sending time is computed as

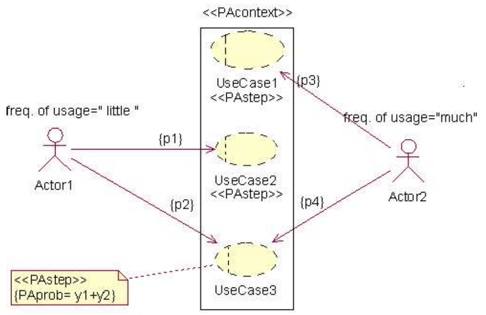

Fig. 6: The use case diagram with performance annotation (fuzzy approach)

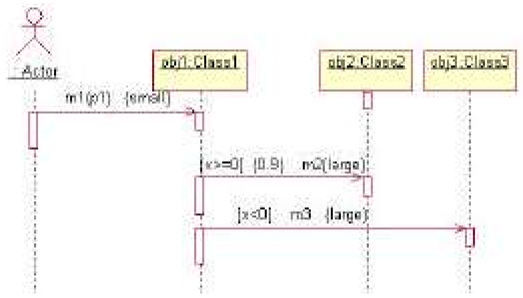

Fig. 7: Sequence diagram with performance annotation (fuzzy approach)

the fuzzy system output. For example from fuzzy rules base, suppose the following fuzzy if – then rule (the fuzzy system is expressed according equation (3)):

If the message size is small and the net speed is high, then the response time is low.

Figure 7 shows the sequence diagram with performance annotation (fuzzy approach). In this figure message m2 has large size and it is sent with probability 0.9.

After drawing the F-UML diagrams with performance annotation, in the next step the diagrams are transformed to the fuzzy formal models (fuzzy Petri Net) and with usage of analysis techniques, the model is solved (system analysis). For more Information about this subject, refer to [4, 5, 9].

THE IMPLEMENTATION OF SUPPOSED SYSTEMS

In this part two case studies are inspected. The first case study is related to the second part of the article and a fuzzy state diagram is explained in an industrial greenhouse. The second case study is related to the third part of the essay and shows the fuzzy sub profile in loan facilities part.

Table 2: Used events and conditions in the fuzzy state diagram

Rule	Event	Condition	Action
R1	-	-	Clear
R2	-	-	A few clear
R3	Season in summer	Pressure_change_velocity is fast	Unstable cloud
	Pressure_change_direction is decreasing		
	Wind_direction is changing to north	Pervious_wind_direction was south	
R4	-	-	Stable cloud

Modeling fuzzy states in an industrial greenhouse: As a case study the system used in an industrial semi-outdoor greenhouse for setting temperature and moisture is studied. Since the greenhouse is semi-outdoor, the weather is effective on its conditions. Every six hours, the weather forecast is done regularly. Considering the weather forecast the rate of temperature and moisture of greenhouse are regulated based on one of the four levels (off, low, medium, high). For example, suppose *unstable cloud* condition is forecasted, so the moisture volume is regulated to low degree. It is considered that, the effectiveness of the regulation of moisture and temperature is work out on system gradually after 6 hours. To implement the supposed system, the fuzzy state diagram for *weather condition* is considered according to Fig. 8.

Table 2 explains the used events and conditions in the above fuzzy state diagram. To see the complete table refer to [5].

A linguistic variable is characterized by a quintuple(Y, T(Y), U, G, M). Linguistic variable *weather condition* is considered as a fuzzy system output with the following features:

T(Y) = {clear, a few clear, unstable cloud, stable cloud}
U= {0, 130}

$$\mu_{clear} = \text{Trapezoidal}(0,0,10,30)$$

$$\mu_{a\ few\ clear} = \text{Trapezoidal}(20,40,50,70)$$

$$\mu_{unstable\ cloud} = \text{Trapezoidal}(60,80,90,110)$$

$$\mu_{stable\ cloud} = \text{Trapezoidal}(100,120,130,130)$$

The supposed fuzzy system possesses *product inference engine*, *singleton* fuzzifier and *center average* defuzzifier (equation (3)). For example, the ultimate output taken after the operation of the defuzzification is equal to "42" (it means fuzzy action is a few clear). In continuation, the matched concrete action is chosen (it means concrete action is medium).So the moisture degree is set on medium. Figure 9 illustrates this subject.

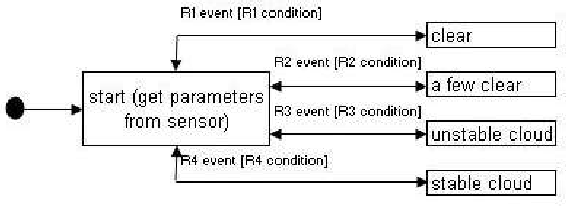

Fig. 8: Fuzzy state diagram for weather condition

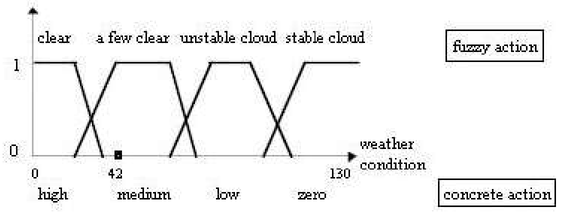

Fig. 9: Regulating greenhouse moisture degree by using the weather forecast

Representation a fuzzy sub profile for the banking system: As a case study, bank system is inspected to receive loan. The receipt of loan in most banks is in this way. Two parameters are effective on loan amount specification: the account opening date and the balance amount. For example, at least n monetary unit should be existed in the account for the specific time (in this case, one year) till loan would be paid to the customer. Now, if the customer withdraw his money before the mentioned time, the loan will not belong to him. This subject comes true in the computation of deposit interest too: if the customer has opened the one year account, he couldn't take his money before the appointed time.

The crisp and certain essence of computations can be improved with the fuzzy sets. By presenting a fuzzy system according to the third equation and by using the fuzzy rules, the amount of allocated loan to the customer is computed. An instance from fuzzy rule base:

if the account balance is little and the amount has been in account for a low time then the loan amount will be little.

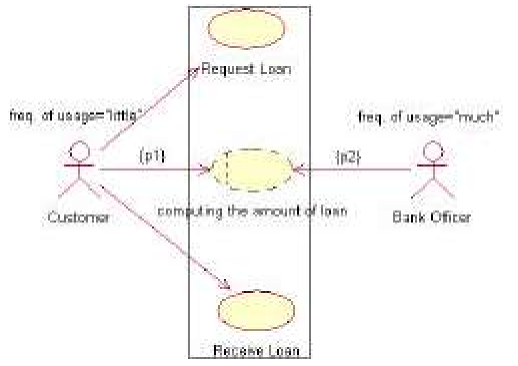

Fig. 10: Use case diagram with tagged value in a fuzzy approach for loan take system

Usage level is one of the performance measures categories. The measures in usage level are intended to evaluate how well the various components of the system are being used. Possible measures are throughput and utilization. Utilization is useful job in time unit. To compute utilization in the above example, the probabilities of the actors (*Bank officer* and *customer*) to execute the use case (*computing the amount of loan*) are computed separately and they sum together (y). Utilization is equal to:

$$U = \frac{TT \times y \times Ar}{Du}$$

Where,
TT is required time for executing the use case,
y is fuzzy system output,
Ar is average of the customer in a day,
Du is duration of work time in a day (by minute)

The implemented system has been done by using the *Rational Software* (version 2003), *Matlab* Tool (for fuzzy computation) and *Pipe2* Tool (for *generalized stochastic Petri Net* implementation). Figure 10 represents the use case diagram and Fig. 2 represents the banking account for this system. In [5, 14] in addition to drawing fuzzy Petri nets, the following cases are specified and the necessary feedbacks for improving the functionality and performance are presented to the system:

- System uncontrollable states (from functionality aspect)
- Different scenarios response time

CONCLUSION AND FUTURE WORK

In this paper, a novel method for development and evaluation of uncertain information systems was explained. The existence of uncertainty in natural problems is a common subject, that F-UML approach is used to investigate it.

The advantages for this method are:

- Uncertainty entering in information systems modeling (from functionality and performance aspects)
- The usage of a standard language and exploitation from its advantages for consistency with software development in different sections
- Investigation to systems structure and behavior aspects
- Defining fuzzy sub profile for supporting the uncertainty without conflicting with UML standard profile (in software performance evaluation section)
- Prevention of complicated fuzzy system output in fuzzy state diagram
- The ability of method to execute on complex systems

Because the idea is new and the usage is abundant, different researches can be defined in this context. Software methodologies that support uncertainty can be propounded as an instance from these researches.

Apart from the above mentioned issue, to enter the propounded idea in object oriented database, Fuzzy - UML could be transformed to Fuzzy – XML. After that the system is evaluated via fuzzy formal models (e.g. fuzzy Petri net).

Setting of information system parameters with intelligent method is one of the cases that can be considered [15].

REFERENCES

1. Object Management Group, 2002. UML Profile for Schedulability, Performance and Time Specification. http:/www.omg.org.
2. Smith, C.U., M. Woodside and P. Clements, 1998. In the Proceedings of the First International Workshop on Software and Performance, Santa Fe, New Mexico, USA, ACM Press.
3. Smith, C.U., 1990. Performance engineering of software systems. The Sei Series in Soft ware Engineering, Addison-Wesley.
4. Haroonabadi, A. and M. Teshnehlab, 2007. Applying Fuzzy-UML for Uncertain Systems Modeling. First Joint Congress on Fuzzy and Intelligent Systems, Ferdowsi University of Mashhad, Iran.

5. Haroonabadi, A., 2006. Design an Intelligent Model in Applying Fuzzy Object Oriented Databases for Systems Development, Ph.D. Proposal, Science and Research Branch, Azad University, Tehran, Iran.
6. Haroonabadi, A. and M. Teshnehlab, 2007. Behavior Modeling in Uncertain Information Systems by Fuzzy-UML. In the Proceeding of the 8th International conference on Computers, Communications and Systems, Daegu University Korea, pp: 51-56.
7. Zongmin, M.A., 2005. Fuzzy information modeling with the UML, Advanced in fuzzy object oriented databases: Modeling and applications, Idea Group Publishing.
8. IBM Company, The Rational Software, version 2003.
9. Merseguer, J. and J. Campos, 2004. Software Performance Modeling using UML and Petri nets. Lecture Notes in Computer Science Journal, 2965: 265-289.
10. Brooks, F.P., 1987. Essence and accidents of software engineering. IEEE, 20 (4): 10-19.
11. Merseguer, J. and J. Campos, 2003. Exploring roles for the UML diagrams in software performance engineering. In the Proceedings of the 2003 International Conference on Software Engineering Research and Practice (SERP'03) (Las Vegas, Nevada, USA), CSREA Press, pp: 43-47.
12. Cortellessa, V. and R. Mirandola, 2000. Deriving a Queuing Network Based Performance Model from UML Diagrams. In Second Intl. Workshop on Software and Performance (WOSP2000), Ottawa, Canada.
13. Bernardi, S., S. Donatelli and J. Merseguer, 2002. From UML Sequence Diagrams and State charts to analyzable Petri Net models. In the Proceedings of the Third International Workshop on Software and Performance, ACM Press, pp: 35-45.
14. Haroonabadi, A. and M. Teshnehlab, 2007. Behavior Modeling in Uncertain Information Systems by Formal Model. In the Proceeding of the Third Conference on Information and Knowledge Technology, Ferdowsi University of Mashhad, Iran.
15. Hayati, M. , B. Karami and M. Abbasi, 2007. Numerical Simulation of Fuzzy Nonlinear Equations by Feed forward Neural Networks. World Applied Sciences Journal 2 (3): 229-234.