

# Transforming Fuzzy State Diagram to Fuzzy Petri net

**H. Motameni**

*Department of Computer Engineering  
Islamic Azad University, Sari Branch,  
Sari, Iran*

E-mail: Motameni@iausari.ac.ir

**I. Daneshfar**

*Department of Computer Engineering  
University of Science and Technology of Mazandaran  
Babol, Iran*

E-mail: Dopofa@yahoo.com

**J. Bakhshi**

*Department of Computer Engineering  
University of Science and Technology of Mazandaran  
Babol, Iran*

E-mail: javadbakhshi@yahoo.com

**H. Nematzadeh**

*Department of Computer Science & Information System,  
University Technology Malaysia*

E-mail: javadbakhshi@yahoo.com

---

## Abstract

*UML is known as one of the most common methods in software engineering. Since this language is semi-formal, many researches and efforts have been performed to transform this language into formal methods including Petri nets. Thus, the operation of verification and validation of the qualitative and non-functional parameters could be achieved with more ability. Since the majority of the real world information is uncertain, therefore fuzzy UML diagram has been extensively used by system analyzers. This paper is an attempt to transform state diagrams created in fuzzy UML into fuzzy Petri net, so that the verification and performance evaluation operation could be performed formally, rather than exact visual analysis.*

**Keywords:** *Software engineering, Fuzzy UML, Fuzzy Petri net, Fuzzy state diagrams.*

---

## 1. Introduction

The Unified Modeling Language (UML) diagrams are extensively used in software design. However, the semi-formal characteristic of this method is a limitation for verification operations and predicting non-functional parameters of the software, especially in the first cycle of the software production. This problem is more critical for control, critical, reactive and real time systems. On the other hand, since the majority of the real world information is uncertain, therefore fuzzy UML diagrams have been extensively used by system analyzers. 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, 2, 3, 4, 5, 6, 7, 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 non-operational parameters and commonly qualitative parameters on the obtained Petri nets of the UML model created [9, 10, 11, 12]. It is obvious that the lack of this important ability in UML models remains the needs of the costumer and the

market unsatisfied. So, this is the reason that makes this type of researches important. In our previous researches [13, 14, 15, 16, 17] besides of studying and presenting transformational patterns for some kinds of usual UML diagrams, especially state diagrams and activity diagrams, 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 state diagrams to fuzzy Petri nets. First, we introduce fuzzy state diagrams briefly. Then, we describe the transform algorithm. At the end, as a case study, we will study the usage of this model for a weather forecasting system.

## 2. Fuzzy UML

UML is known as one of the most important tools in extending object oriented systems. This language makes visual modeling possible so that the system developers will be able to standardize and make understandable the ideas and establish a more effective mechanism in relations with other patterns [18, 19]. In a proposed general pattern since the real world information is mostly uncertain, in many cases these types of information cannot be modeled by UML. Recently a model named fuzzy UML, has been introduced [20,21,22] which has the UML characteristics, and is also able to model uncertain concepts.

### 2.1. Fuzzy state diagrams

State diagrams, models different states of an object. This diagram is mostly used to show the dynamic behaviors of a system. Figure 1 shows an example of a state diagram.



*Figure 1. An example of a state diagram*

A state diagram is formed of five sections which are :

- 1- Start state
- 2- Different states of an object life cycle
- 3- Events
- 4- Guard conditions
- 5- End state

As shown in Figure1, the start state is displayed with a solid black circle with a symbol which refers to an initial state of an object in its life cycle. The end state is displayed with a double circle symbol which refers to the end state in this cycle. In a state diagram there is only one start state, whereas, the end state can be omitted or there can be several end states.

In a state diagram each state is displayed by a rectangle which shows the different states of an object in its life cycle.

An event transforms the object from one state to another. Guard condition which is binded in brackets, controls the occurrence of a transition.

According to these explanations, a fuzzy state diagram is a graphical model in fuzzy UML which shows the different states of a fuzzy object in its real world life cycle.





This diagram uses fuzzy rules for transforming the state of an object to another state. A fuzzy rule is shown as below

```
<on event list <event threshold>>
  if condition list <EC coupling>
  Then action
```

Fuzzy rules are used to show the real world rules for an object in which these rules can be active or deductive. As an example, the above mentioned rule is an active one. If the on part is omitted, then it becomes a deductive rule. If in the on part, the threshold is omitted in active rules, the threshold is assumed to be an exact matching with a value of 1.

Each section of the state diagram can be transformed to a fuzzy state diagram. Table 1 shows these transformations, clearly.

*Table 1. Transformation of a state diagram into its fuzzy state*

State Diagram	Fuzzy state Diagram
	
	
state	Action of rule
[condition]	[Fuzzy condition]
Event	Fuzzy Event

### 3. Fuzzy Petri nets

We introduce the following fuzzy Petri net (FPN) [26] structure to model the fuzzy rules:

$(P, P_s, P_e, T, T_F, TRTF, A, I, O, TT, TTF, AEF, PR, PPM, TV)$ , 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)  $PPM : P \rightarrow PR$ , 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.

A token value in place  $p_i \in P$  is denoted by  $TV(p_i) \in [0, 1]$ . If  $TV(p_i) = \mu_i$ ,  $\mu_i \in [0, 1]$  and  $PPM(p_i) = d_i$ . This states that the degree of the truth of proposition  $d_i$  is  $\in \mu_i$ . A transition  $t_i$  is enabled if  $\forall p_i \in I(t_i), \mu_i > 0$ . If this transition  $t_i$  is fired, tokens are removed from input places  $I(t_i)$  and a token is deposited onto each of the output places  $O(t_i)$ . Since we provide parameter passing, the token value of an output place  $p_k \in O(t_i)$  is calculated from that of the input places  $I(t_i)$  using the transition function  $TF_i$ , where  $TF_i = TRTF(t_i)$ . This token's membership value to the place  $p_k$ , (i.e.,  $\mu_k = TV(p_k)$ ), is part of the token and gets calculated within the transition function  $TF_i$ , where  $\mu_k = TF_i(I(t_i))$ .

#### 4. Transformation algorithm

Before studying the meaning of transformation Algorithm it is necessary to introduce the meaning of scenario. Scenario is a parameter that can divide the rules. Only one of the states of this parameter can be active at a time. The substitution of the scenario is specified by the user. In the fuzzy deduction cycle of the strength of event  $e$  for rule  $r$  in scenario  $s$  is calculated with formula [26]

$$Strength(e, r, s) = m_s(r) * m_{ef}(value(e_c)) \quad (1)$$

Which uses scalar multiplication. Where value( $ec$ ) is the value of the event (fuzzy or crisp) occurred,  $\mu_{ef}$  is the membership function of the fuzzy event  $ef$  and  $\mu_s(r)$  is the similarity of the rule  $r$  to the current scenario  $s$ . The formula of  $\mu_s(r)$  is defined as

$$m_s(r) = \max([\min(\max(m(A_s, A_r), \max(m(C_s, C_r))) * RLV_{rs} / RLV_{\max})] \quad (2)$$

Where  $A_s \in S, \forall A_r \in R_i, C_s \in S, \forall C_r \in R_i, RLV_{rs}, RLV_{\max} \in S$ . In that  $A$  and  $C$  correspond to the antecedent and consequent of a rule. The antecedent is composed of event and condition whereas the consequent is composed of action/conclusion. Here  $A_s$  is the antecedent of a current scenario meta-rule and  $A_r$  is the antecedent of  $R_i$  (the rule

to be evaluated),  $C_s$  is the consequent of a meta rule in the scenario and  $C_r$  is the consequent of the rule to be evaluated,  $RLV_r$  is the relevance value of the meta-rule to the current scenario and  $RLV_{max}$  is the maximum of those relevance values.

The Fuzzy UML state diagram created will be transformed to a fuzzy Petri net according to the steps below.

Step1. First for each state change in this diagram, its event and conditions must be found. Suppose a state diagram as Figure2.

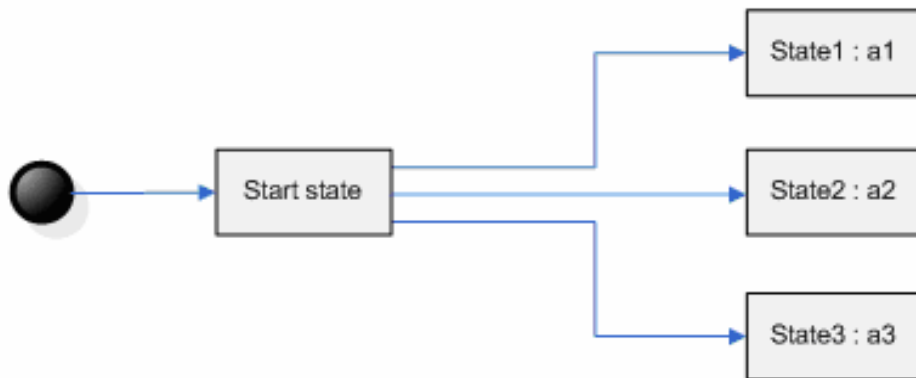


Figure 2. An example of a state diagram

The events and conditions calculated for the state diagram in Figure4 is represented in table 2

Table 2. Rules applied on state diagram in Figure4

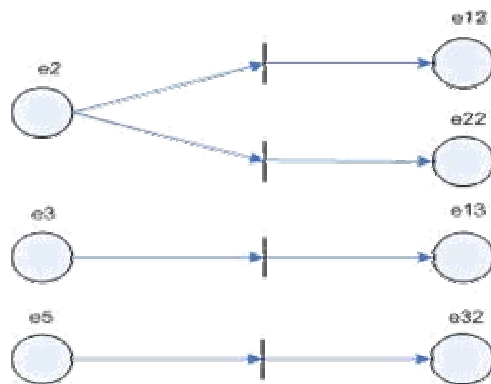
Rule	Event	Condition	State
R <sub>1</sub>	e <sub>1</sub> is e <sub>11</sub>	C <sub>1</sub>	a <sub>1</sub>
	e <sub>2</sub> is e <sub>12</sub>		
	e <sub>3</sub> is e <sub>13</sub>	C <sub>2</sub>	
R <sub>2</sub>	e <sub>1</sub> is e <sub>21</sub>	C <sub>1</sub>	a <sub>2</sub>
	e <sub>2</sub> is e <sub>22</sub>	C <sub>2</sub>	
		C <sub>3</sub>	
R <sub>3</sub>	e <sub>1</sub> is e <sub>31</sub>	C <sub>1</sub>	a <sub>3</sub>
	e <sub>5</sub> is e <sub>32</sub>		

Step2. The highest level of divisions in the rules concluded is found. As Sean in table 2, e<sub>1</sub> is in the condition part of all rules so e<sub>1</sub> is selected as the scenario. So, the rules are classified according to the scenario in table3.

**Table 3.** Classification of the rules according to the scenario

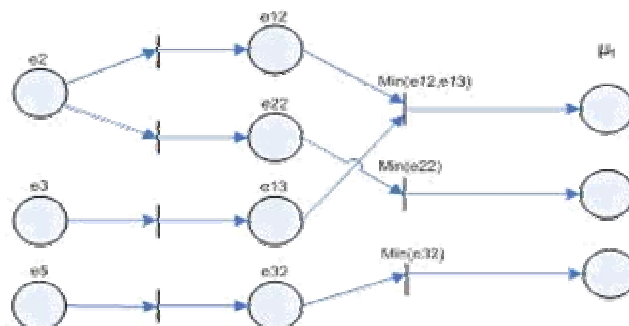
Scenario	Rule	Event	Condition
e <sub>11</sub>	R <sub>1</sub>	e <sub>2</sub> is e <sub>12</sub> e <sub>3</sub> is e <sub>13</sub>	C <sub>1</sub> C <sub>2</sub>
e <sub>21</sub>	R <sub>2</sub>	e <sub>2</sub> is e <sub>22</sub>	C <sub>1</sub> C <sub>2</sub> C <sub>3</sub>
e <sub>31</sub>	R <sub>3</sub>	e <sub>5</sub> is e <sub>32</sub>	C <sub>1</sub>

Step3. For each parameter defined in all the rules we create a place where these Parameters can't be repeated and also can't be a scenario parameter. Then for different kinds of states which these parameters can accept in all of the rules, we create a place. These places are jointed to the proper places with a transition, as shown in Figure3.



**Figure 3.** Step 3 in transition algorithm

Step4. For each rule a transition is placed and the events of each rule are applied on the transition and we place the min function on the transition, where the value of this function is the value of  $\mu_{ef}$  for each rule, Figure 4.



**Figure 4.** Step 4 in transition algorithm

Step5. To calculate the strength of each event on the specified rule in an active scenario, first we have to calculate  $\mu_e$  value of  $\mu_s(r)$  using the formula below.

$$m_{s(r)} = \max(< \min(\max(m(A_s, A_r)), \max(m((c_s, c_r)) * RLV_{rs} / RLV_{\max} >): \tag{3}$$

For each rule we create a transition which one of its inputs is a place which is initialized by the value  $\mu_s(ri)$  and the other input of the transition is the previous output which is the value  $\mu_{ef}$  and the output of the transition which is another place which holds the effective Value.(Figure5)

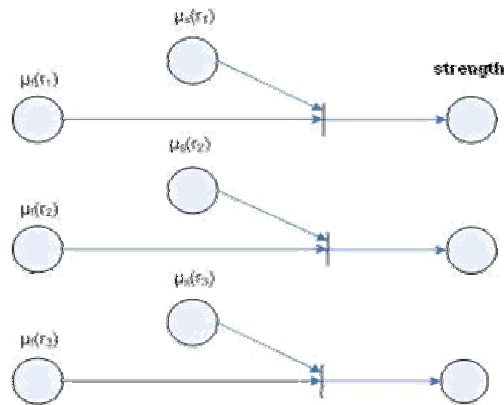


Figure 5. Step 5 in transition algorithm

Step6. We create a place for the condition of each rule and we valueate each condition with the fuzzy values calculated. (Figure 6)

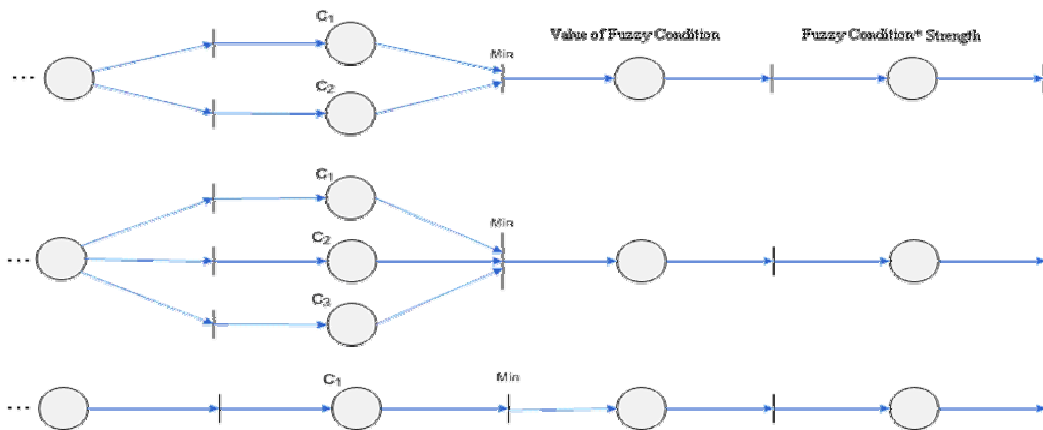


Figure 6. Step 6 in transition algorithm

Step7. Now, we apply the result of this FPN which is the states of the state diagram to the FPN. (Figure 7)

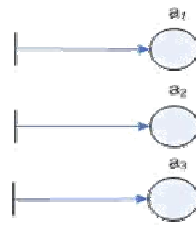


Figure 7. Step 7 in transition algorithm

Lemma. If the state diagram is as shown in Figure8, for drawing the fuzzy Petri net for state3 and showing its reliability to state2, we must place state2 as the condition of state3 in the fuzzy Petri net, which its value is calculated from the value of the fuzzy Petri net from state2.

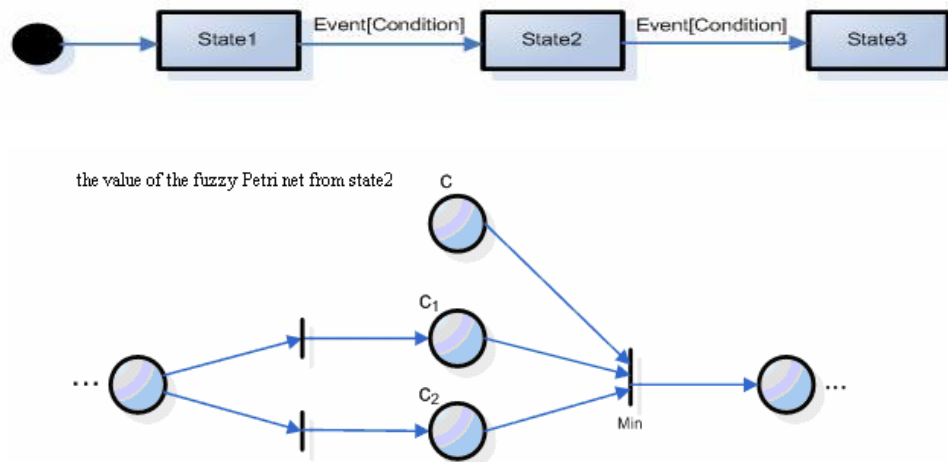


Figure 8. A part of the fuzzy Petri net in state3

### 5. Case study

The case study that we are going to study is a part of a weather forecasting system. In this system the atmospheric elements are pressure, temperature, humidity, wind and cloudiness. This system has two kinds of forecasting. In the first one, which is the expected weather, it can be one of clear, clear few, clouded instable or clouded stable. In the second kind, we determine the expected weather event according to the output of the first part together with the newly changing parameters on the atmospheric elements. In the second level expected weather event can be any of rain, shower, snow, hail or fog.



The UML diagram in Figure9 shows the classes related to this system and the relationships between each class. Where, the Expected\_Weather class contains indexes which by changes in these parameters the class object changes to one of the weather conditions: clear, clear few, clouded instable or clouded stable. The next class is the Expected\_weather\_Event which its object shows one of the weather conditions. The purpose of index in the explanations above is the amount and the size of changes, direction changing and speed changing and etc. The third class is the season class shows the type of the season that we are in.

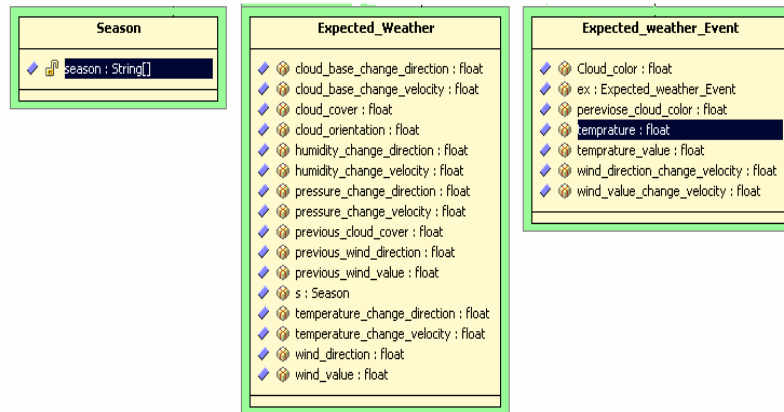


Figure 9. An example class diagrams

The fuzzy state diagrams available in Figure10 and Figure11 shows the state diagram of Expected\_weather and Expected\_weather\_Event classes. As seen, both classes object first go to start state. After receiving the weather condition from the sensor, the object will be brought to a possible state according to the fuzzy rules.

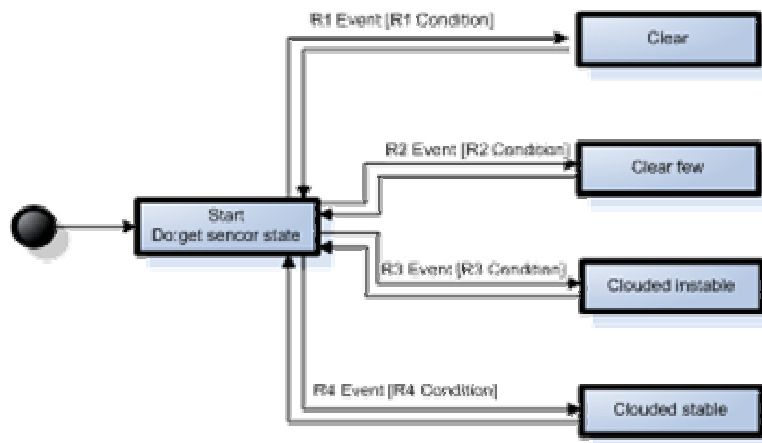


Figure 10. State diagram for the Expected\_Weather class

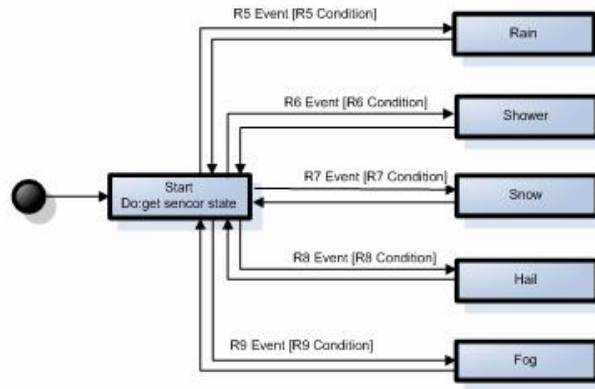


Figure 11. State diagram for the Expected\_Weather\_Event class

Table 4, 5 shows the fuzzy rules which change the state of an object to another state for each state diagram.

Table 4. The events and conditions used in state diagram in Figure 10

Rule	Event	[Guard Condition]	Action
R <sub>1</sub>	Season is [Summer] wind_direction is changing to [north OR northwest], humidity_change_direction is decreasing cloud_cover is changing to broken sky cloud_base_change_direction is increasing	pressure_change_velocity is fast previous_wind_direction was south humidity_change_velocity is fast previous cloud_cover was overcast cloud_base_change_velocity is fast	Clear
R <sub>2</sub>	Season is [Summer] pressure_change_direction is increasing cloud_cover is changing to [broken sky OR few] cloud_base_change_direction is increasing humidity_change_direction is decreasing wind_direction is changing to [north OR northwest]	pressure_change_velocity is slow wind_value is breeze previous_wind_direction was [south OR southwest] previous_cloud_cover was [overcast OR cloudy]	Clear Few
R <sub>3</sub>	pressure_change_direction is decreasing humidity_change_direction is increasing wind_value is changing to [medium_strong OR strong] cloud_orientation is changing to vertical cloud_base_change_direction is decreasing temperature_change_direction is increasing	pressure_change_velocity is fast humidity_change_velocity is fast previous_wind_value was [breeze OR medium_strong]	Clouded Instable
R <sub>4</sub>	pressure_change_direction is decreasing humidity_change_direction is increasing temperature_change_direction is increasing cloud_base_change_direction is decreasing cloud_orientation is changing to horizontal	pressure_change_velocity is slow humidity_change_velocity is slow temperature_change_velocity is slow cloud_base_change_velocity is slow wind_direction is [southeast OR south]	Clouded Stable

**Table 5.** The events and conditions used in state diagram in Figure 11

Rule	Fuzzy event	[Fuzzy Guard Condition]	Action
R <sub>5</sub>	expected_weather is clouded_stable wind_direction is changing to [south OR southwest] wind_value is changing to medium_strong cloud_color is changing to grey	Previous_wind_direction was north wind_direction_change_velocity is slow previous_wind_value was breeze wind_value_change_velocity is slow previous_cloud_color was white,	Rain
R <sub>6</sub>	expected_weather is clouded_instable cloud_color is changing to dark_grey	Previous_wind_value was calm Previous_cloud_color was grey	Shower
R <sub>7</sub>	expected_weather is clouded_stable wind_direction is changing to north cloud_color is changing to grey	temperature is below 0 Celsius degrees previous_wind_direction was south Previous_cloud_color was white	Snow
R <sub>8</sub>	expected_weather is clouded_instable cloud_color is changing to dark	temperature is very_high previous_wind_value was breeze Previous_cloud_color was grey	Hail
R <sub>9</sub>	expected_weather is clouded_stable	wind_value is [calm OR breeze]	Fog

According to the fuzzy state diagram and the association's relations between the classes, table-6 will be obtained.

**Table 6.** The classifications of rules

Season	Expected – Weather	Expected – Weather Event
Summer	Clear, Clear few	--
Winter	Clouded stable	Rain, hail, shower
Spring	Clouded instable	Rain, hail, shower
Fall	Clouded instable	Rain, hail, shower

It is deducted from the table-6 that season can be selected as the largest division of the sections. So season can be a scenario.

Now using the transformation algorithm the created state diagrams the fuzzy Petri net is deducted Figure12.

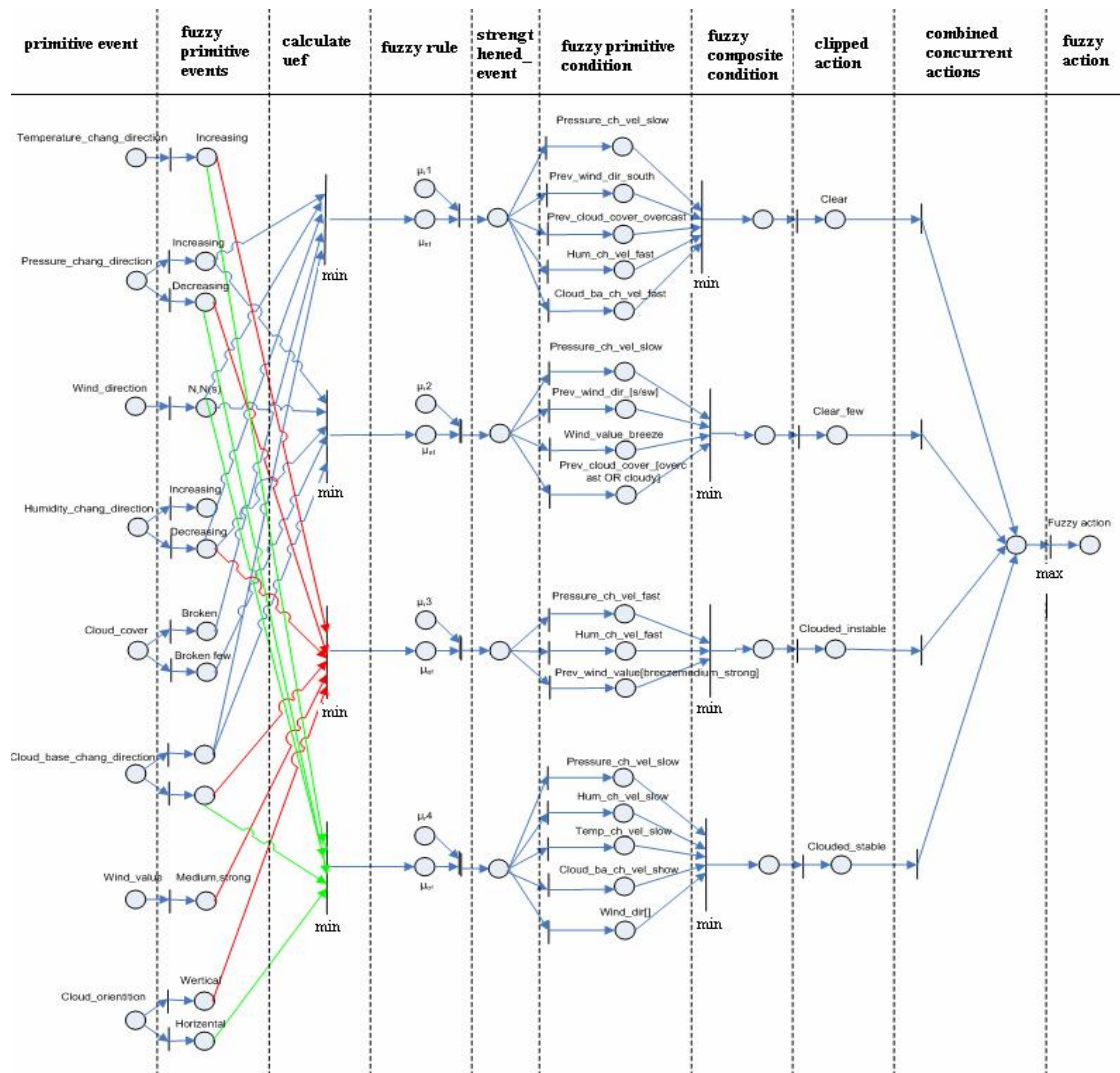


Figure 12. The Petri net for the Expected\_Weather\_Event state diagram

Now we consider that we are in winter season and the sensor of parameter condition is in the time of T1 and T2 shows the values in table 7

Table 7. The variables in t1 and t2

Attribute	t1	t2
Wind value	6	6.5
Cloud orientation	0	22.5
Cloud base	7200	1200
Wind direction	255	210
Humidity	50	57
Temperature Value	-4.5	1-
Pressure value	1003.5	1000
Cloud cover	2	3

Table-8 shows the fuzzy values of the events and conditions of each rule.

**Table 8.** The fuzzy values of the condition and event parts according to Table-7

Attribute	R <sub>1</sub>	R <sub>2</sub>	R <sub>3</sub>	R <sub>4</sub>
Wind value	-/-	-/-	/-0.5	-/-
Cloud orientation	-/-	-/-	/-0.5	-0.5/
Cloud base	/-0	/-0	/-1	0.5/1
Wind direction	/-0	/-0	-/-	0.66-/
Humidity	/-0	/-0	/-1	0.5/1
Temperature Value	-/-	-/-	/-1	/-1
Pressure value	/-0	/-0	/-1	0.5/1
Cloud cover	/-1	/-1	-/-	-/-

In the first step we want to study how the fuzzy Petri net for the Expected\_weather class works. And according to the values given and events and conditions display how the object of this class changes. For this propose we act as below.

1- Fuzzify the events for each rule and calculate the value of  $m_{EF}(ri)$ .

:R<sub>1</sub>

$$\mu_{\text{Pressure-ch-dir-inc}}(-3.5) = 0$$

$$\mu_{\text{Wind-dir-n/nw}}(210) = 0$$

$$\mu_{\text{hum-ch-dir-dec}}(7) = 0$$

$$\mu_{\text{Cloud-cover-broken-sky}}(3) = 1$$

$$\mu_{\text{Cloud-basech-dir-dec}}(-5600) = 0$$

$$\Rightarrow m_{ef}(R_1) = \min(0, 0, 0, 1, 0) = 0$$

:R<sub>2</sub>

$$\mu_{\text{Pressure-ch-dir-inc}}(-3.5) = 0$$

$$\mu_{\text{Cloud-cover-broken-Few}}(3) = 1$$

$$\mu_{\text{Cloud-base-ch-dir-inc}}(-5600) = 0$$

$$\mu_{\text{Hum-ch-dir-dec}}(7) = 0$$

$$\mu_{\text{Wind-dir-n/nw}}(210) = 0$$

$$\Rightarrow m_{ef}(R_2) = \min(0, 1, 0, 0, 0) = 0$$

:R<sub>3</sub>

$$\mu_{\text{Pressure-ch-dir-dec}}(-3.5) = 1$$

$$\mu_{\text{Hum-ch-dir-inc}}(7) = 1$$

$$\mu_{\text{Wind-value-ms/s}}(-0.5) = 0.5$$

$$\mu_{\text{Cloud-base-ch-dir-dec}}(-6000) = 1$$

$$\mu_{\text{Temp-ch-dir-inc}}(3.5) = 1$$

$$\Rightarrow m_{ef}(R_3) = \min(1, 1, 0.5, 1, 1) = 0.5$$

:R<sub>4</sub>

$$\mu_{\text{Pressure-ch-dir-dec}}(-3.5) = 1$$

$$\mu_{\text{Hum-ch-dir-inc}}(7) = 1$$

$$\mu_{\text{Temp-ch-dir-inc}}(3.5) = 1$$

$$\mu_{\text{Cloud-base-ch-dir-dec}}(-6000) = 1$$

$$\mu_{\text{Clouded-orientation-horizontal}}(75) = 0.5$$

$$\Rightarrow m_{ef}(R_4) = \min(1, 1, 1, 1, 0.5) = 0.5$$

As seen R<sub>3</sub>, R<sub>4</sub> can be fired because their value of  $\mu_{EF}$  is greater than 0. So the active parts of the fuzzy Petri net are the routes relating to these two transitions.

2- Calculating the strength of the events.

We calculate the strength of an event in a scenario using the formula

$$\text{Strength}(e, r, s) = m_{ef}(r_i) * m_s(r_i)$$

So, the strength value for R<sub>3</sub>, R<sub>4</sub> is as below

$$R_3 \Rightarrow \text{strength}(e, R_3, s) = m_{ef}(R_3) * \mu_{\text{Winter}}(R_3) = 0.5 * 0 = 0$$

$$R_4 \Rightarrow \text{strength}(e, R_4, s) = m_{ef}(R_4) * \mu_{\text{Winter}}(R_4) = 0.5 * 1 = 0.5$$

So, only the rule  $R_4$  can continue its activity in the rest of the fuzzy Petri net and the  $R_3$  deactivates.

### 3 - Fuzzificating the condition part of the active rules.

Calculate the MEF for the conditions as calculated for the events in step1. Because  $R_4$  is the only active rule we will calculate MEF for  $r_4$ .

$$\mu_{\text{Pressure-ch-velocity-slow}}(-3.5) = 0.5$$

$$\mu_{\text{hum-ch-velocity-slow}}(7) = 0.5$$

$$\mu_{\text{temp-ch-velocity-slow}}(3.5) = 0.5$$

$$\mu_{\text{Cloud-base-velocity-slow}}(-5600) = 0.5$$

$$\mu_{\text{Wind-dir-s/sw}}(210) = 0.66$$

$$\Rightarrow \min(0.5, 0.5, 0.5, 0.66) = 0.5$$

So  $R_4$  can still continue its activity.

### 4- Find the clipping value

Up to this point only  $r_4$  succeed its antecedent matching degrees  $0.5 * 0.5 = 0.25$

So clipping value of 0.25 is used for the action part of  $r_4$  which is an expected weather of clouded-stable

### 5- Find the state of the object

Maximum of the clipping values for each active rule in the fuzzy Petri net diagram which for the only active rule  $R_4$  is:

$$\max(0.25) = 0.25$$

Which means an expected weather forecasted for the state of the object of class Expected\_Weather which fuzzy condition and events is clouded stable

Figure 13 shows the steps used.

## 6. Conclusion

In this paper with the purpose of formalization of the state diagrams in fuzzy UML for a stronger verification and validation of qualitative parameters in the analysis model created by analyzers, an algorithm is represented to transform the fuzzy state diagram into fuzzy Petri net. Also for a case study a weather forecasting system is studied which has an applied aspect.

Researchers for a further work decide to extend this algorithm for different types of fuzzy UML diagrams specially activity diagrams and design a software engineering tool so it can automatically perform the transformation operation.

Verification and applying the validation of efficiency on the result fuzzy Petri nets will be the future work for designers.

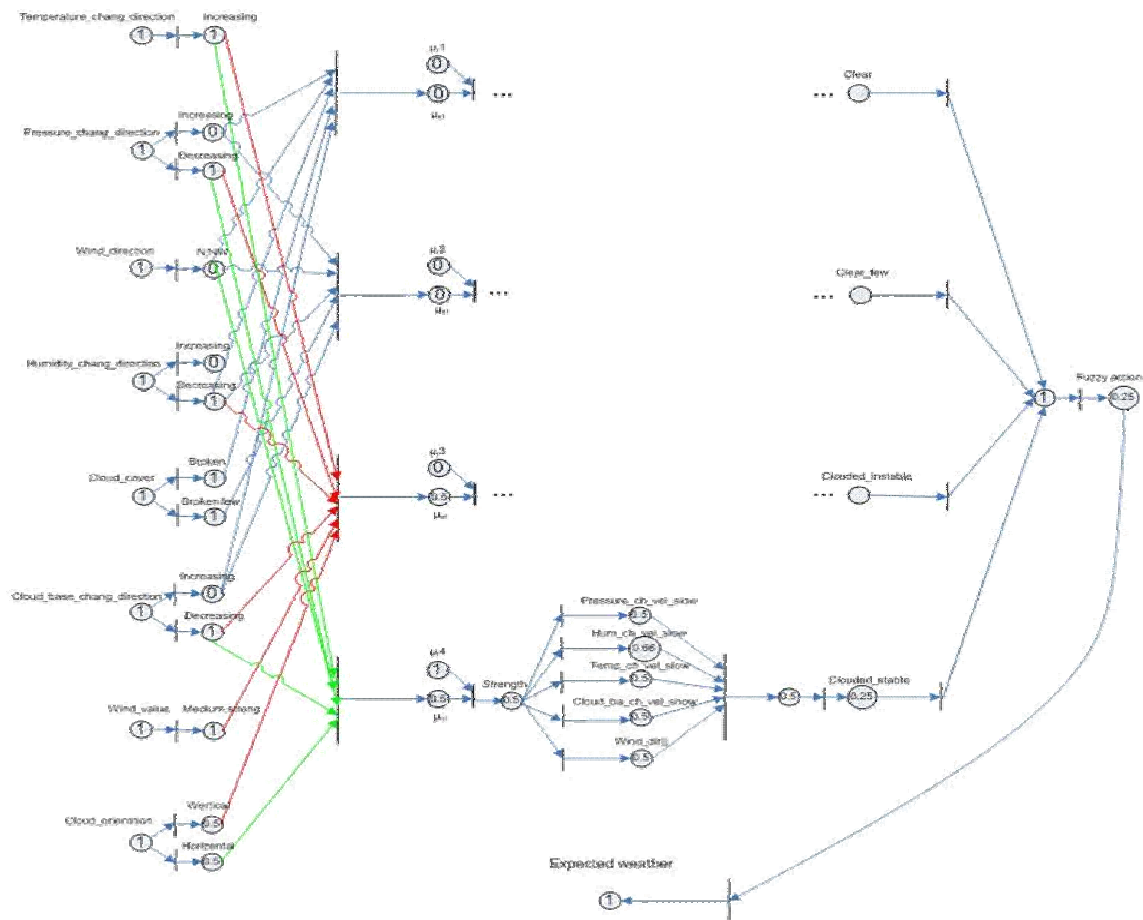


Figure 13. The fuzzy Petri net of class Expected\_Weather running

## References

- [1] Faul M. B. "Verifiable Modeling Techniques Using a Colored Petri Net Graphical Language" Technology Review Journal, spring/summer, 2004.
- [2] Shin, M., Levis, A. and Wagenhals, L. "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.
- [3] Bernardi, S. Donatelli, S. and Merseguer, J. "From UML Sequence Diagrams and Statecharts to Analysable Petri Net Models" ACM Proc. Int'l Workshop Software and Performance, pp. 35-45, 2002.
- [4] Eshuis, R. "Semantics and Verification of UML Activity Diagrams for Workflow Modelling" Ph.D. Thesis, University of Twente (2002).
- [5] Pettit, R. G. and Gomaa, H. "Validation of dynamic behavior in UML using colored Petri nets' UML" (2000), Zaragoza, Spain (2002) 295-302.
- [6] Saldhana, J. and Shatz, S. M. "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).
- [7] Elkoutbi, M. and Rodulf K. Keller: "Modeling Interactive Systems with Hierarchical Colored Petri Nets" 1998 Advanced Simulation Technologies Conf., Boston, MA (1998) 432- 437.
- [8] L. Bernardinello, F. De Cindio, "A Survey of Basic Net Models and Modular Net Classes", LNCS, vol. 609, Springer-Verlag, 1992, p.609.

- [9] Balsamo, S. et al *"Model-Based Performance Prediction in Software Development: A Survey"* IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. 30, NO. 5, MAY 2004 p295.
- [10] Merseguer, J. , L'opezGrao, J. P., Campos J. *"From UML Activity Diagrams To Stochastic Petri Nets:Application To Software Performance Engineering"* ACM, WOSP 04 January 1416, 2004,
- [11] Fukuzawa, K. 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.
- [12] Merseguer, J., Bernardi, S., Campos, J. and Donatelli, S. *"A Compositional Semantics for UML State Machines Aimed at Performance Evaluation"* M. Silva, A. Giua and J. M Colom (eds.), Proc. of the 7th Int. Workshop on Discrete Event Systems (WODES'02), Zaragoza, Spain (2002) 295-302.
- [13] Motameni,H,et al,Mapping State Diagram to Petri net: *"An Approach Tousemarkov Theory For Analyzingnon-Functional Parameters"* ieee,international conferenceon computer,information and system science,december 4\_14 2006 , university of bridgport, USA(presented).
- [14] Motameni,H,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,moscow, russi, (presented).
- [15] Motameni, H., Zandakbari, M. and Movaghar, *"Deriving Performance Parameters From the Activity Diagram Using Gspn and Markov Chain"*, ICCSA 2006 Proceeding of 4<sup>th</sup> International Conference On Computer Science and Its Aapplications, San Ddiego, California, 2006.
- [16] Motameni, H et al. *"Evaluating UML State Diagrams Using Colored Petri Net"* SYNASC' 05.
- [17] Motameni, H et al. *"Verifying and Evaluating UML Activity Diagram by Converting to CPN"* Proc of SYNASC'05,Romania,Sep 2005, (presented).
- [18] Object Management Group *"UMLTM Profile for Schedulability, Performance, and Time Specification"*OMG Document, Version 1.1, January 2005.
- [19] Rumbuaugh,j.,Blaaha,m.,Premarlani ,W.,Eddy,F.,Lorensen,W.(1991). *"Object-Oreinted Modeling And Design "*, prentice hall , Englewood Cliffs,nj,USA.
- [20] Wang lu, *"Fuzzy UML"*,Seminararbeit,Sommersemester 2005.
- [21] Zongmin Ma. *"Fuzzy Information Modeling With the Uml"*. Idea, 2005.
- [22] Z.M. Ma(2004). *"Extending UML For Fuzzy Information Modeling In Object\_Oriented Database "*, theories and practices,idea group publishing.
- [23] T. Murata, Petri Nets: *"Properties, Analysis And Applications"*, Proceedings of IEEE 77 (1989) 540–541.
- [24] L. Bernardinello, F. De Cindio, (Ordinary) Petri Nets (PN), <<http://www.daimi.au.dk/PetriNets/classification>>.
- [25] K. Jensen, *"Colored Petri nets (CPN) "*, <http://www.daimi.au.dk/PetriNets/classification/level3/CPN.html>>.
- [26] Burcin Bostan-Korpeoglu,Adnan Yazici ,A Fuzzy Petri Net Model For Intelligent Database, Data & Knowledge Engineering (2006), Elsevier,2006.