Management Science Letters 3 (2013) 2681–2690

Contents lists available at GrowingScience

Management Science Letters

homepage: www.GrowingScience.com/msl

A new method for performance evaluation of enterprise architecture using streotypes

Samaneh Khamseh[a*], Fares Sayyadi[b] and Mohammad Hosein Yektaie[c]

[a]*Department of Computer Engineering, Science and Research Branch, Islamic Azad university, Ilam, Iran*
[b]*Faculty Member of Electrical and Science, Science and Research Branch, Islamic Azad university, Ilam, Iran*
[c]*Faculty Member of Computer and Science, Abadan branch, Islamic Azad university, Abadan, Iran*

C H R O N I C L E

Article history:
Received May 12, 2013
Received in revised format
12 August 2013
Accepted 28 September 2013
Available online
October 8 2013

Keywords:
Enterprise Architecture
Redundancy
Fuzzy UML
Stereotypes
Reliability
Fuzzy petri nets
Execution model

A B S T R A C T

These days, we see many organizations with extremely complex systems with various processes, organizational units, individuals, and information technology support where there are complex relationships among their various elements. In these organizations, poor architecture reduces efficiency and flexibility. Enterprise architecture, with full description of the functions of information technology in the organization, attempts to reduce the complexity of the most efficient tools to reach organizational objectives. Enterprise architecture can better assess the optimal conditions for achieving organizational goals. For evaluating enterprise architecture, executable model need to be applied. Executable model using a static architectural view to describe necessary documents need to be created. Therefore, to make an executable model, we need a requirement to produce products of the enterprise architecture to create an executable model. In this paper, for the production of an enterprise architecture, object-oriented approach is implemented. We present an algorithm to use stereotypes by considering reliability assessment. The approach taken in this algorithm is to improve the reliability by considering additional components in parallel and using redundancy techniques to maintain the minimum number of components. Furthermore, we implement the proposed algorithm on a case study and the results are compared with previous algorithms.

1. Introduction

These days, most organizations use the recent advances of information technology for making strategic decisions. Many organizations have complex infrastructure with improper architecture, low efficiency, flexibility and speed to transfer the information. Enterprise architecture is a set of representations or models described in connection with a description of an organization, and the primary objective is to manage and use necessary items. Architecture includes a large number of documents, which describe all parts of organizations (Deft, 2000). The problem with this description is on how they all be noted and be used. Therefore, to create order and organization of the enterprise

*Corresponding author. Tel:+989365903223
E-mail address: Samaneh_khamseh@yahoo.com (S. Khamseh)

doi: 10.5267/j.msl.2013.10.012

architecture description, a framework needs to be applied. There are some shortcomings on modeling notation to cover all C4ISR products and enterprise is one of the most important challenges facing the C4ISR architecture framework. The necessity of such a modeling notation where the use of a variety of symbols and language modeling for cover crops, causing confusion and inconsistency and architecture work is hard and complicated. The task of mapping products is normally accomplished using Enterprise Architecture Framework C4ISR, which is established by unified modeling language (UML) as an object-oriented approach (Sowa & Zachman, 1992). However UML is unable to express the needs of vague and indeterminate instances because the needs of users for information systems is based on the formation, Therefore, access to some of the system's requirements is a challenging task. In order to overcome these shortcomings, the Fuzzy-UML is implemented (Zadeh, 1983; Bostan-Korpeoglu & Yazici, 2006). In any enterprise architecture, the process name, process enterprise architecture and the three-phase development strategy, planning, architecture and implementation of the architecture need to be clearly specified. According to Lindsay et al. (2003) analysis of enterprise architecture planning phase can be accomplished to evaluate the behavior and the performance evaluation. Since most software systems are unable to handle all necessary needs in evaluating enterprise architecture, to evaluate these kinds of systems they must first create an executable model. In this paper, to do this, a fuzzy colored Petri nets are proposed. There are many ways to create an executable model of the software and its evaluation. However, the primary objective of this paper is based on an execution model using stereotypes fuzzy UML diagrams. To evaluate the behavior of enterprise architecture, the fuzzy UML diagrams are used. However, in this paper, to create executable models of software systems, the existing stereotypes in the use case diagram, sequence and deployment are applied. One of the most important parameters implemented to evaluate a software system is associated with reliability of a system and a system with high level of reliability ensures long-term performance. In other words, reliability is defined as the probability that a system will work, properly under pre-defined circumstances. In this paper, by using an executable model of the stereotypes created by the three fuzzy UML diagram above, we improve reliability of the system (Kaisler et al., 2005).

The remaining paper is organized as follows: The second part of the paper is devoted in enterprise architecture description. In the third part of the paper, we propose a method to evaluate the reliability of enterprise architecture. The fourth section of the paper evaluates the results and findings of the proposed algorithm implemented on a case study and the results are addressed. Finally, in the fifth section of the paper, conclusions and future work are described.

2. Literature review

There are various techniques to evaluate the architecture and the aim of this paper is to investigate the effect of architectural styles, which is essential that the techniques based on the models and simulations used to evaluate the architecture. To use this technique, it is necessary to first develop a model to analyze and to evaluate architectures. Both models are mainly for work done: First, the architecture products are designed, under certain modeling language, and products and architecture, are commonly created by standard modeling language like UML. Next, using these products, we create a model for evaluating the architecture. For the ARCHIMATE model, architecture is evaluated in terms of performance. With the normal practices of the organizational model and its parameters, this model is suitable for quantitative analysis of architecture. For the proposed model, we can quantitatively analyze the architecture, the language, the characters and relationships are added to the context in which certain quantities can be determined. There is a method developed by Levis (Shin et al., 2003), which creates an executable model and the main advantage of this method is that Architecture Modeling Language UML is implemented in this model, which is a popular technique. In this way, the enterprise architecture implemented C4ISR architecture products are produced in this context and colored Petri nets are applicable for this methodology (Wagenhals et al., 2003). OSAN is one of the powerful language modeling for evaluating the architecture, especially the architecture performance evaluation. Bai et al. (2008) applied a model of the architecture caused. The advantage

of this modeling language is that the model fully supports object-oriented programming. In recent years, much attention is devoted on evaluating enterprise architecture. JavadPour and Shams (2009) used C4ISR as an architecture framework with regard to the fixed frame as architectural structure, to evaluate the performance of the software architecture with different behaviors (different styles) with Network Colored Petri Nets. Mozaffari et al. (2011) performed an investigation on enterprise architecture to analyze and to evaluate enterprise architecture knowledge of software architecture design, to achieve appropriate architecture. Rezaei and shams (2009) presented a solution, which extracts the enterprise architecture federal enterprise architecture framework. They also defined the maturity level of the enterprise architecture that includes a detailed assessment of the existing architecture. Here is emphasize is on the uncertain nature of the system providing a technique for increasing the reliability of the system (Behbahaninejad et al., 2012).

3. The proposed method

This article focuses on assessment of the performance of enterprise architecture using stereotypes UML. UML is a standard language for describing semi-formal enterprise architecture, which is easy to address functional needs and to address uncertainty we use fuzzy terms and develop F-UML (Ma, 2005; 2011b). Since UML is not a formal model, evaluation of software systems is not possible, directly and we need to have the access to the actual model. In this case, for the structural aspects of system, we use case diagram, sequence diagram and deployment diagram to show the behavior of the physical aspects of the system resources (Bernardi & Merseguer, 2007). The proposed solution using stereotypes performance in F-UML is suitable for modeling and evaluating the performance of enterprise architecture. Fig. 1 shows details of our implementation.

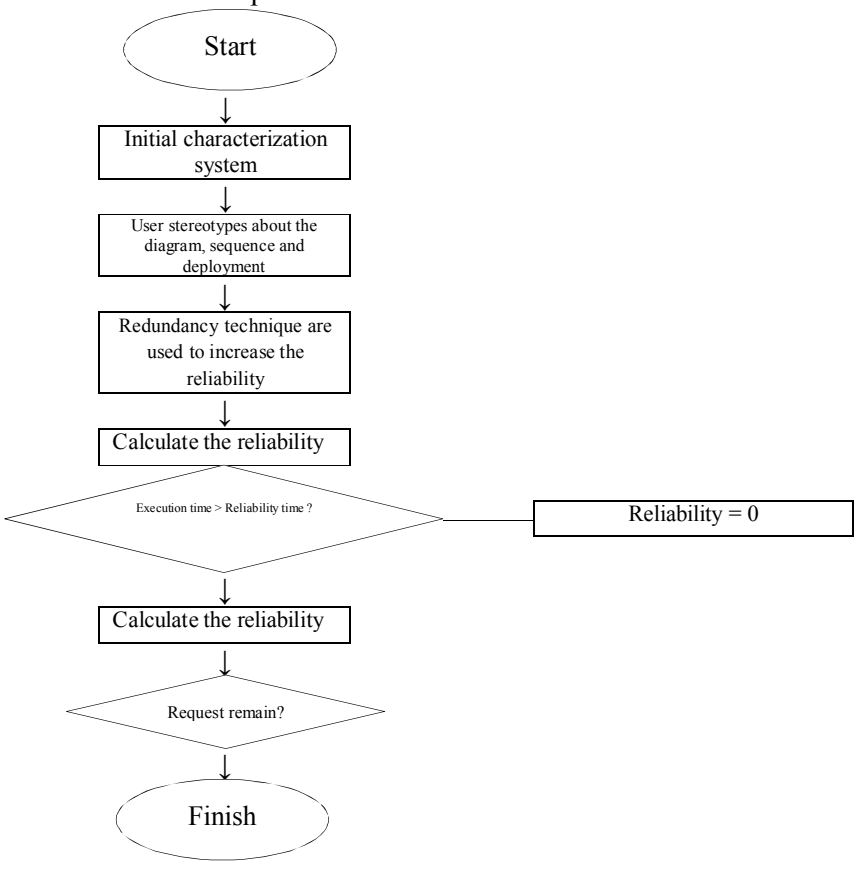

Fig. 1. The flowchart of the proposed algorithm

3.1 Stereotypes used in use case

Generally, any user on the use case represents a sequence of requests in the system. This graph has the following stereotypes:

1- <<PAopenload>>: Used in cases where the request sequence is infinite that Tags <<PAoccurrence>> that is, the time between two consecutive requests shows.
2- <<PAcolsedload>>: Used in cases where the request sequence is limited and contains the following tag:

1-2- PApopulation: Shows the total number of requests in the system.
2-2- PAextDelay: Full time interval between a request and subsequent interaction with the system show.

For the proposed algorithm, the use case diagram and label the stereotype << PAclosedload >> PApopulation, PAextDelay, are used.

3.2. Stereotypes used in sequence diagrams

In this diagram, all existing interactions in the system are displayed. To add efficiency considerations on the sequence diagram, the stereotype << PAstep >> is used. These stereotypes include all of the following labels on system reliability assessment, which are used:

1- Label size: It specifies the size of the message.
2- Label demand: Rate will apply to the supply of services.
3- Label PAhost: The name is a reference to the requested resource.
4- PAprob: Indicates the likelihood of the message.
5- PArep: Indicates the message is repeated.

3.3. Stereotypes used in deployment diagrams

Deployment diagram explains how to get a picture of the physical system resources. In this diagram, for additional performance information, the stereotype << PAhost >> uses labels that include the following:

1- Label PArate: Shows the processing rate.
2- Label Schdpolicy: Policy schedule shows.

In our proposed algorithm, to assess the reliability of the system, we will also use these two tags.

To enhance system reliability, the proposed algorithm uses the redundancy technique. To this end, each message is assigned to more than one component for parallel execution. Therefore, if one component fails during execution, by replacing the components of the same source, we will prevent the failure process and, thereby, we will increase system's reliability.

To add redundancy to the system, only two stereotypes, PArate, PAdemand, will be manipulated. Thus, if we have *n* elements from a source system, then the deployment diagram, *n* will have to PArate and graphs can be arranged from *1* to *n* to do pAdemand. The following figure illustrates this better than the words:

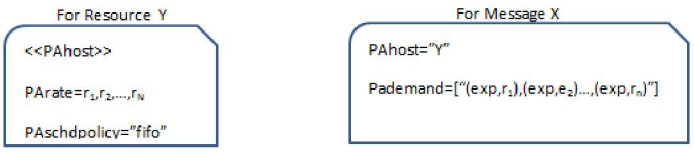

Fig. 2. The proposed stereotypes with tags

We need to calculate reliability:

If the CPU processing rate is rpp (PArate), service request rate from the source is rpd (PAdemand) name, the first step is to calculate the service rate as follows:

$$SR[i,X] = \frac{rpp[i]}{rpd[i]} \tag{1}$$

Order of SR[I,x], the service rate of component *i* in the *X* source.

The service rate for each component of the resources at run time to get the message size *z* with *y* (size) of component *i*, going to X source is calculated as follows,

$$T[i,X,y] = \frac{y}{SR[i,X]} \tag{2}$$

To the T[*i, x, y*], *y* is the size of the message by the time i^{th} component of *x* as a source.

Since the proposed algorithm increases the reliability of the system, using the redundancy technique, the simulations is carried out and each message may be processed by several sources. Let $T_{x,r}$ represents the minimum time of the message *x,* therefore we have,

$$T_{x,r} = \min[T(i,X,y)] \qquad 1 \le i \le n \tag{3}$$

For the execution of a task, all messages must be executed, so the time to run messages, to run the greatest time of task execution time will be considered. However, if the messages are dependent on each other, the total running time of all times, as summarized in the task. Namely,

$$T = \begin{cases} \max(T_{x,r}[i] & 1 \le i \le m \quad \text{if no correlation} \\ \sum_{i=1}^{m} T_{x,r}[i] & 1 \le i \le m \quad \text{else} \end{cases} \tag{4}$$

A completion of a task cannot be longer than logged and system reliability can be calculated as follows:

$$R_{T^*} = \sum_{i=1}^{I} P_i . l(T_i < T^*) \tag{5}$$

We represent the entire task entered into the system. Let T^* be the same as $T_{reliability}$. T_i computation time to run the task with the number I as follows,

L(true)=1, L(false)=0 (6)

Let P_i be the number of running task, which is calculated as follows,

$$P_i = \prod_{j=1}^{m} q_j \tag{7}$$

where q_j denotes the probability of the message with the number *j* of task *i*, which will be calculated as follows:

$$q_j = p(e)^{bp} * p(d) \tag{8}$$

where P (e) is the possibility that *e* constituents working during processes, *P (d)* is the probability that communications channels is likely to be healthy, *bp* is the number of components and *e* is busy status.

In our proposed algorithm, if a message is performed by multiple sources, to calculate the reliability we have:

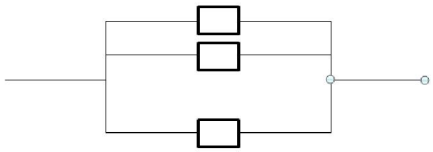

Fig. 3. Parallel system with resource

$$Reliability = 1 - \prod_{j=1}^{n} (1 - R_j) \tag{9}$$

6. Case study

We now apply the proposed algorithm to measure the reliability for some case studies. As an example, an automated teller machine (ATM), due to its complexity and the possibility of Product Architecture Framework C4ISR is considered. After describing the problem, the executable product model architecture (diagrams UML) is created. Then, using the proposed method and simulation CPN tools, we calculate the applicable reliability. An ATM with another entity, the Client (User) and the Bank is the interaction. One of the main operations of customers in the system operation is associated with "withdraw money" transaction. In this case, first, the user has to insert bankcard into the ATM system. The bank, the validity of the card, and ATM banking system and the password are requested from the customer. If the card is invalid, the system retrieves the customer's bank ATM card, otherwise the credibility of the customer is the key. If the password is invalid, the system returns the card to the customer. Otherwise, it shows the available system options such as cash withdraw and transfer funds. Customer-options "withdraw money from the account" is selected and the system will issue the requested withdrawal amount. This is a message that customer enters to withdraw some amount money and the system checks whether there is enough money in the account or not. When there is insufficient amount of fund, the system gives a warning to customer, otherwise, a bank ATM, withdraw amount is deducted from customer's account. Finally, the client application and the back end of the card, the customer can return the card system. As stated earlier, the pattern of sequence diagram between components, i.e. the interactions among different components are plotted. The following sequence diagram of use case "withdraw money" shows.

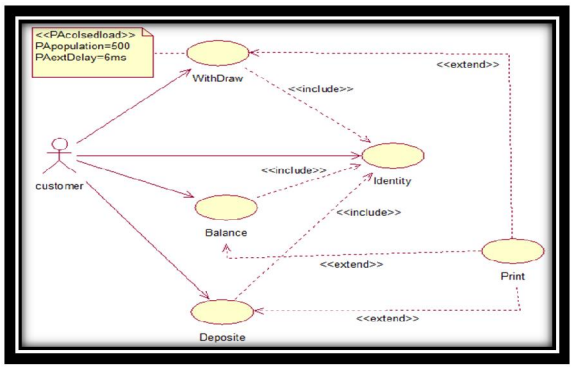

Fig. 4. Use case of ATM

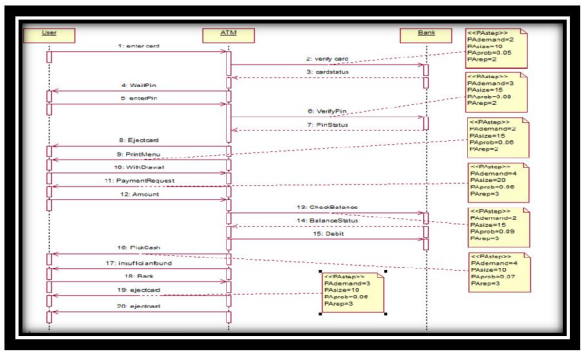

Fig. 5. Sequence Diagram for ATM

Fig. 6 shows deployment diagram for ATM.

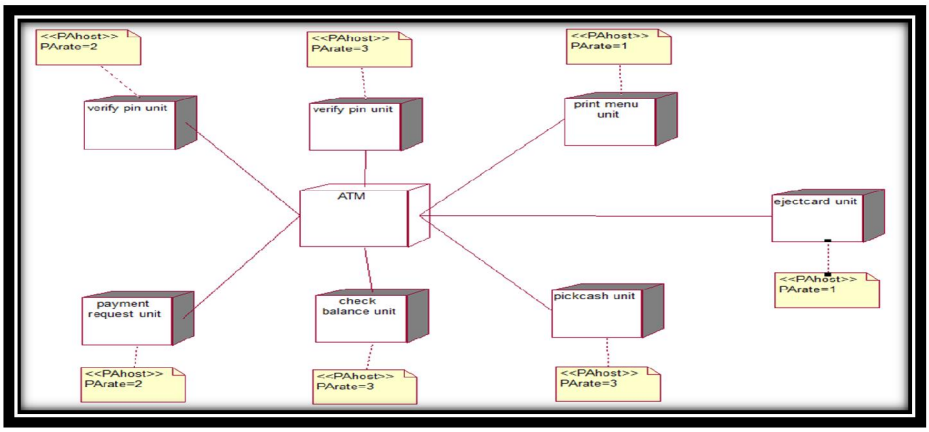

Fig. 6. Deployment Diagram for ATM

Fig. 7-11 demonstrates details of our implementation to create an executable model of a home page CPNTOOLS

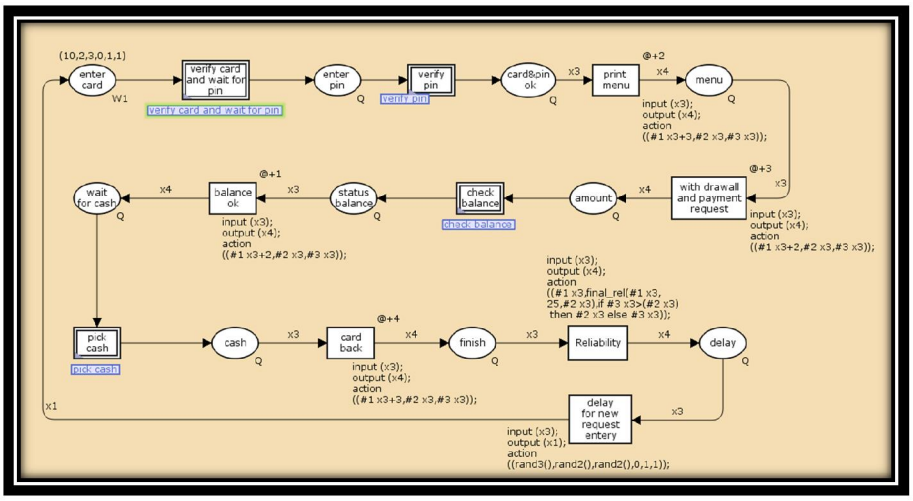

Fig. 7. The high-level model of ATM

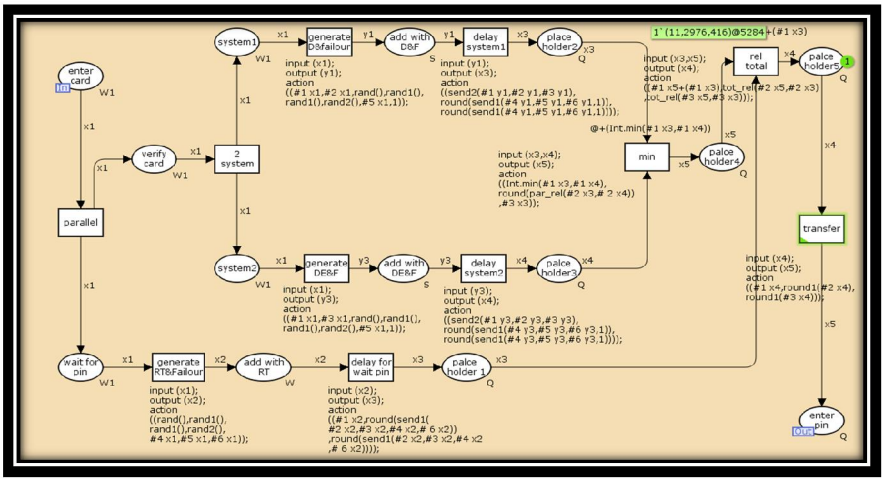

Fig. 8. Subpage OF verify card and wait for pin

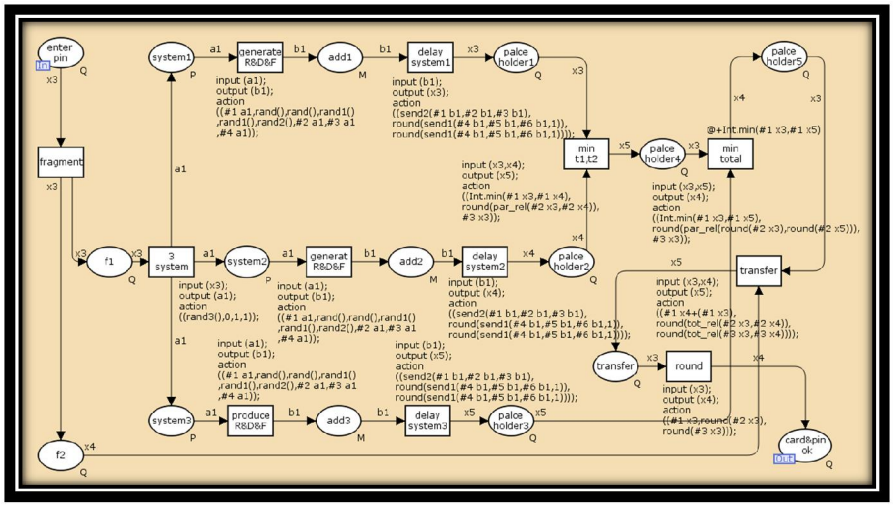

Fig. 9. Subpage for wait pin

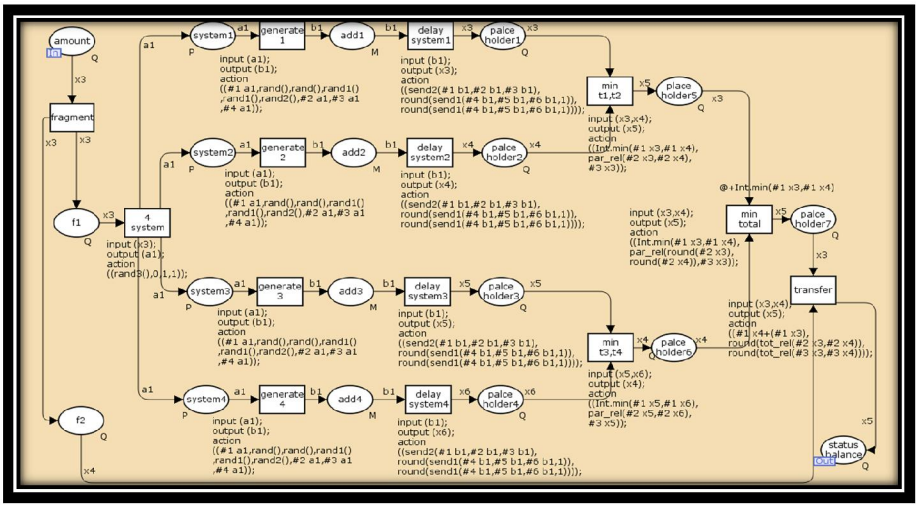

Fig. 10. Subpage for check balance

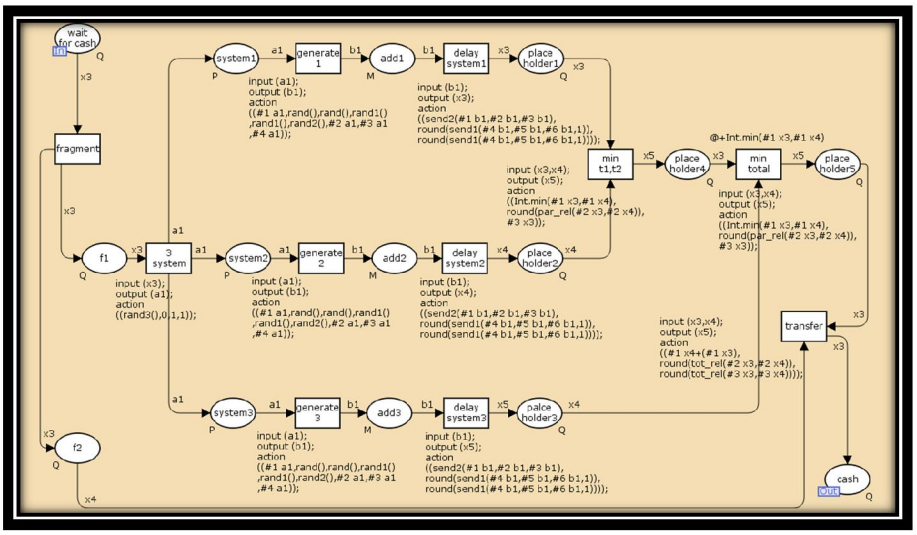

Fig. 11. Subpage for Pick cash

However, after creating an executable model of sequential phase diagram and phase diagram of the user, the system calculates the reliability. Fig. 12 demonstrates the reliability computed by our proposed algorithm and comparison the reliability with an existing method.

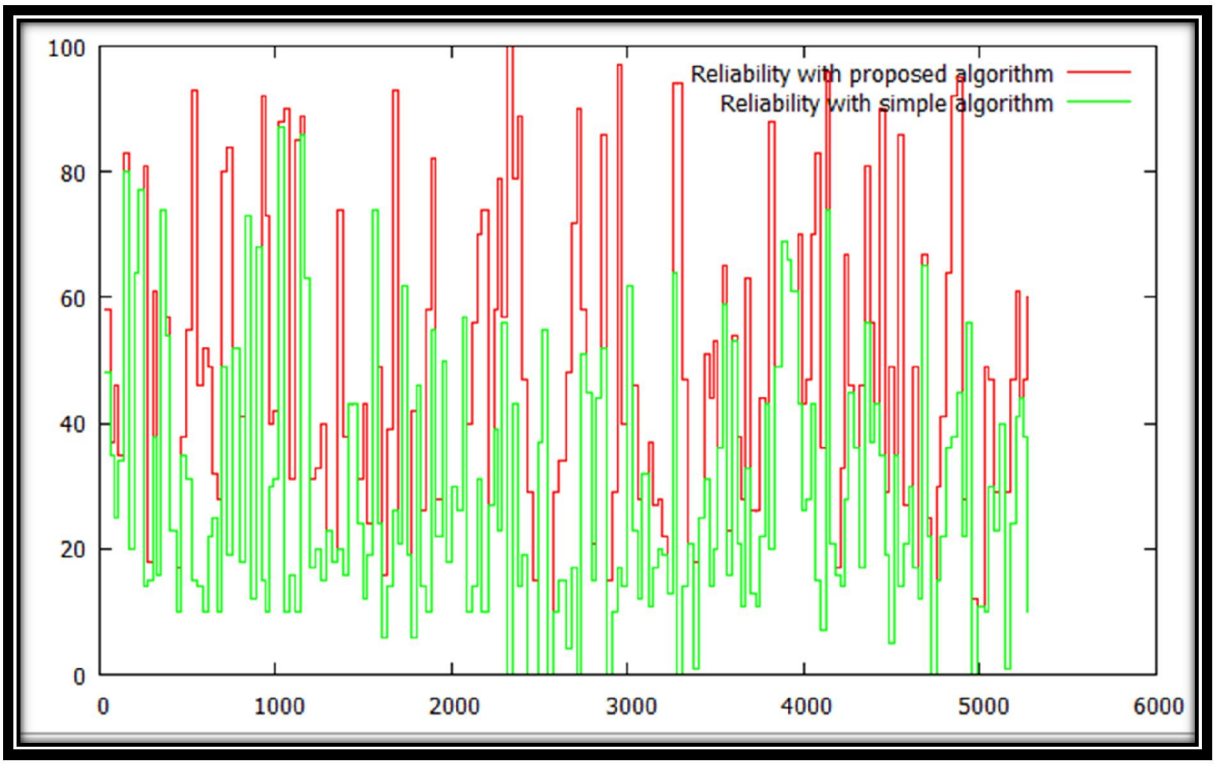

Fig. 12. Compared proposed method with previous method

As we can observe from Fig. 12, using the redundancy in the system increases the reliability of the system, significantly.

7. Conclusion

In this paper, an algorithm was presented in the diagrams using stereotypes of F-UML and the reliability of the system was measured. We first evaluated the performance of the stereotypes and some diagrams including use case and sequence were used for the deployment using the art of F-UML technique. The performance of the proposed algorithm was compared with other algorithms. The redundancy technique has been employed to increase the reliability of the system. After creating an executable model, their algorithm on a case study has been implemented. The results indicate that the proposed algorithm provides more reliable results compared with other previous algorithms.

Acknowledgment

The authors would like to thank the anonymous referees for constructive comments on earlier version of this paper.

References

BaI, X.H. (2008). An application with UML object-based Petri Nets for C4ISR architecture simulation validation. *proceeding of a seventh international conference on machine learning and cybernetics*, Kunming.

Behbahaninejad, P., Harounabadi, A., & Mirabedini, S.J. (2012). Evaluating software architecture using fuzzy formal models. *Management Science Letters*, 2, 469–476.

Bernardi, S., & Merseguer, J. (2007). Performance Evaluation of UML Design with Stochastic Well-Formed Nets. *The Journal of Systems and Software*, 80, 1843–1865.

Bostan-Korpeoglu, B., & Yazici, A. (2006). A fuzzy Petri net model for intelligent database. *Data & Knowledge Engineering*, 8, 112-122.

Deft, R. (1983). Organisation theory and design. *New York: West*.

Rezaei, R., & Shams, F. (2009). Providing a comprehensive method for developing and evaluating enterprise architecture plan. *The first Conference on Enterprise Architecture in Practice*, Isfahan, Iran.

Javadpour, R., & Shams, F. (2009). Performance evaluation of electronic city architecture using colored Petri nets. The 2nd Conference on Electronic City, Tehran, May 2009.

Kaisler, S. H., Armour, F., & Valivullah, M. (2005). Enterprise architecting: Critical problems. Proceeding of the 38th Hawaii International Conference on system sciences, 8(8), 224.2.

Ma, Z. (2005). Fuzzy Information Modeling With the UML. *Idea Group Publishing*, 153-176.

Ma, Z.M., Zhang, F., & Yan, L. (2011a). Fuzzy information modeling in UML class diagram and relational database models. *Applied Soft Computing*, 11, 4236-4245.

Ma , Z.M. , Yan , L., & Zhang , F. (2011b). Modeling Fuzzy information in UML class diagrams and object oriented database models. Fuzzy Sets & Systems, 186, 26-46.

Lindsay, A., Downs, D., & Lunn, K. (2003). Business processes—attempts to find a definition. *Information and software technology*, *45*(15), 1015-1019.

Shin, M. E., Levis, A. H., & Wagenhals, L. W. (2003). Transformation of UML-based system model to design/CPN model for validating system behavior. In *Proc. of the 6th Int. Conf. on the UML/Workshop on Compositional Verification of the UML Models*.

Sowa, J. F., & Zachman, J. A. (1992). Extending and Formalizing the Framework for Information Systems Architecture. *IBM Systems Journal*, 31(3), 590-616.

Wagenhals, L. W., Haider, S., & Levis, A. H. (2003). Synthesizing executable models of object oriented architectures. *Systems Engineering*, *6*(4), 266-300.

Zadeh, L.A. (1983). The role of fuzzy logic in the management of uncertainty in Expert System. *Fuzzy Sets Systems*, 11, 199-227.