International Review on Computers and Software (I.RE.CO.S.), Vol. 10, N. 10
ISSN 1828-6003 *October 2015*

Evaluating Open Source Software Usability Using a Multistage Fuzzy Model Approach

Thamer A. Al-Rawashdeh

Abstract – *In recent years, development of Open Source Software has obtained significant importance in the production of software products. Although, developers of Open Source Software have developed software with functional competitiveness as compared to closed proprietary software; computer users still prefer closed proprietary software than open source due its usability strength.*
On the other hand, once the usability of OSS is evaluated, it would be easier to develop and implement an acceptable and qualitative product, since the software usability is considered to be one of the most important quality factors. Thus, this work proposed a multistage fuzzy model approach for evaluating the Open Source Software usability, which includes nine usability characteristics to be taken into account when designing and implementing OSS software. The model takes a project, developed in MATLAB and quantifies its usability. The Analytical Hierarchy Process (AHP) technique was employed to verify the proposed model approach and to rank its usability characteristics. These characteristics are sequenced according to its importance as follows; learnability, understandability, efficiency, error prevention, memorability, operability, familiarity, attractiveness, and usability-compliance. ***Copyright © 2015 Praise Worthy Prize S.r.l. - All rights reserved.***

***Keywords**: Open Source Software, Analytical Hierarchy Process, MATLAB, Fuzzy Model, Software Usability, Usability Characteristics*

Nomenclature

FURPS	An acronym that represents a model to classify attributes of software quality including Functionality, Usability, Reliability, Performance, and Supportability
ISO/IEC 9126	An international standard for evaluating software quality
ISO 9214-11	A multi-part standard from International Organization for Standardization (ISO) for evaluating software usability
ISO 14915	An international standard for GUI designing
ISO 16071	An international standard for accessibility of human-computer interface
HFES/ANSI 200	Human Factors Engineering of Software User Interfaces (HFES 200) from American National Standards Institute (ANSI)
MATLAB	A high-level computing language for data visualization, data analysis, and algorithms development
ω	Eigenvector of the characteristics' matrix
λ	Eigenvalue, $> n$

I. Introduction

Open Source Software (OSS) is defined as a computer software with its source code that is available and licensed to be used, modified and enhanced by anyone [1]-[46]. This is granted under the GPL – General Public License, which allows users to freely customize and modify various source codes available across the web, as per their requirements. Primarily, it is developed in a public, and a collaborative manner [1].

Developed OSS applications can be as influential as the proprietary software, i.e. Mozilla Web Browser, Open Office Software, Linux Operating System etc. [2].

Fitzgerald (2006) [20] maintains that, with the involvement of and acceptance from large commercial IT vendors, OSS products transitioned from a fringe activity into the mainstream software culture. However, it is reality that Open Source Software are often plagued with poor usability [2]-[4]. End users of many popular Open Source Software products shared comments that reveal blend praise of its functionality with criticism of its usability [3].

Additionally, Hall (2014) [5] indicated that the OSS usability is often ignored in the development stage. Since, most of the Open Source Software projects focus on the software functionality than the user interface [5].

In this respect, poor usability evaluation causes failure of the software systems that leads to staff unproductivity, user dissatisfaction, a substantial monetary loss, and time waste [4]. On the other hand, Software usability is an important factor in quality assessment of a software product. Therefore, if the usability of Open Source Software is evaluated, it would be easier to develop an acceptable and qualitative products. With the passage of time, this concept is gaining importance and being adopted by individuals, small and large enterprises.

There have been several attempts to evaluate the usability of software in the past [6]-[11]. Most of these studies have considered the software usability evaluation after the software implemented and deployed. Some other researches have provided usability principles as set of intermediate artifacts with regard to the compatibility to a set of guidelines.

Since the traceability between such artifacts and final software has not been established well. Solving this problem lies in using a model-driven engineering approach due the realistic traceability mechanism.

Therefore, the work has been done to propose a multistage fuzzy model approach to evaluate and quantify the usability of OSS [6]. This includes factors that should be considered when designing and implementing an OSS.

It gathers three main contributions including: the conceptual model that provides requirements for developing an OSS with desired usability; adapting and introducing nine usability characteristics in the OSS context; and early usability evaluation through quantifying the usability characteristics of conceptual model. Software usability is defined as the degree to which a product can be used by particular users to achieve specified goals with effectiveness, efficiency and satisfaction in a given context of use [12].

It could be extracted from previous definition that, there are several usability characteristics that have affected the software usability [13]. Therefore, in order to propose a model in order to evaluate usability of Open Source Software; and to clarify the contributions of this study, the literature on software quality and software usability has been conducted in the next section.

II. Literature Review

As previously mentioned, in this section the literature on software quality and usability has been reviewed, in order to propose a model to evaluate the usability of OSS. In the field of software quality and software usability measurement such related issues such as usability related concepts, usability characteristics, and usability sub-characteristics, there is one point on which researchers generally agree: there is no well-defined model that has been proposed to evaluate and quantify software usability [9], [13], [20].

Open Source Software shares an effective and simple development model, which grows on the basis of a community of developers who share their application codes, leading to expansion of this concept in software usability.

Moreover, it is not limited to a set of development concepts, as pretty much desktop, web and server applications are utilizing this medium.

Well-known quality models including McCall, Boehm, FURPS, Dromey, and ISO/IEC 9126 presented the software usability as an important characteristic in improving software acceptability and reliability, and user satisfaction [16], [43], [44], [45]. These quality models have also provided various sub-characteristics to measure the software usability e.g. understandability, learnability, operability, attractiveness, and usability compliance [13]-[16].

Eason (1984) [17] classifies the software usability in a structured set of characteristics and sub-characteristics as follows: task characteristics, system characteristics, and user characteristics. Where, the task characteristics category has been decomposed into two attributes including frequency and openness [17]. System characteristics category consists ease of learning, ease of use, and task match. Finally, User characteristics category includes discretion, motivation, and knowledge. Shackel (1991) [18] illustrated the importance of usability engineering and relativity of its concepts.

He measured the usability of software through effectiveness, learnability, attitude, and flexibility. Later, Nielsen (1993) [19] introduced the software usability as an important characteristic that affect the software acceptance. He mentioned that the software usability is too abstract for a term to be directly evaluated.

Therefore, five sub-characteristics had been given learnability, efficiency, memorability, errors, and satisfaction. ISO 9214-11 provides three sub-characteristics to evaluate the software usability, including effectiveness, efficiency, and satisfaction. Table I summarizes the characteristics that used to measure the software usability. In addition to the fuzzy of usability characteristics, the generality of previous quality models means that further analysis and mapping of usability characteristics is required.

TABLE I
USABILITY CHARACTERISTICS

Characteristics	Eason model	Shackle model	Nielsen model	Quality models
Frequency	X			
Openness	X			
Knowledge	X			
Motivation	X			
Discretion	X			
Ease of learning	X			
Ease of use	X			
Task match	X			
Effectiveness		X		X
Learnability		X	X	X
Flexibility		X		
Attitude		X		
Efficiency			X	X
Memorability			X	
Errors			X	
Satisfaction			X	X
Understandability				X
Operability				X
Attractiveness				X
Usability compliance				X

This leads to the usability characteristics cannot be measured directly, but it should be evaluated in terms of characteristics objective and software [17]. This includes, the context of evaluation should be taken into consideration, before determining on specific usability characteristics to be used [21], [22].

In the same regard, according to Benson et al., 2004 [27] the OSS usability related issues are not the same as those for closed proprietary software. Therefore, the proposed fuzzy usability model describes the features; of Open Source Software and the usability characteristics that are related to such features.

III. Characteristics Affecting OOS Usability

Existing quality models provide usability characteristics that are general and common for evaluating the usability of every type of software product. However, each software type has its own characteristics [19]. Therefore, the existing usability characteristics should be modified and extended according to the nature of a new software being evaluated. This includes, redefining some characteristics, eliminating others, and introducing new usability characteristics.

Therefore, totally nine usability characteristics have been refined and introduced in this section according to the Open Source Software nature and its features. Eight usability characteristics are adapting: learnability, understandability, efficiency, memorability, error-prevention, operability, attractiveness, and usability compliance; and new one has been introduced namely software familiarity.

Learnability is defined as the capability of software to enable users to learn the main software functionality and to obtain proficiency to complete a task [23]. In Open Source Software context, the learnability could be evaluated by measuring the time a user needs to perform a task in the time an expert would take to complete the same task.

Understandability refers to the capability of a software product to enable the user to understand whether the software is suitable, and how it can be used for particular tasks and conditions of use [24]. In this respect, in addition to the purpose of Open Source Software, all of the design and user documentations must be clearly written, so that it is easy to understand.

Efficiency is described as the number of tasks that a user can perform using a software per unit [25], [26]. It could be measured by the task completion rates and task completion time. The more efficient Open Source Software is the faster users can perform a task and complete a task. This is among the major characteristics, as a reason of this concept gaining popularity.

Memorability is related to the capability of a software to enable users to use the software when they return to it [14]. The importance of memorability for Open Source Software is some Open Source Software i.e. operating systems (Linux) are quite complicated, so the high memorability software enables users to use such applications when they come back to them. The memorability characteristic is influenced by variety factors. For instance, an action makes the users have a reaction, if this reaction leaves the users with a good feeling they will not forget it.

The *Errors Prevention* characteristic refers to the number of errors a user makes while performing a task rather than the software [27]. Low error rate while Open Source Software being used implies a good usability. On the contrary, high error rates, a low efficiency and satisfaction [28]. In the same respect, proper design of error message is an important topic in software usability.

Therefore, many factors should be taken into account to make such error messages helpful to the users. Among these factors are the error message should exist, the message should understandable, the message should be precise, the message should be correct, and the messages should be consistent. Furthermore, error prevention improvises time consumption while the development processes [29].

The *Operability* characteristic is related to the capability of software to enable the user to operate and control it [14]. There are many factors that affect the software operability: suitability of the software components for a task, self-descriptiveness of the software components, the controllability of the software components, the conformity of the software components with users' needs, error tolerance of the software components, and suitability of the software components for individualization [14]. Therefore, all of these factors should be considered while OSS is being developed.

Attractiveness refers to the capability of software to be attractive to its users [30]. According to Maedche eat al., 2012 [30] the software attractiveness has been influenced by pragmatic and hedonic factors perspicuity, up-to-datedness, aesthetic, simulation, and novelty.

Thus, the user interface of Open Source Software including windows, menus, bars, messages… etc. should be consistent, aesthetic and visually attractive. Additionally, it is worth to mention that, due to the accessibility of Open Source Software code, users would have an ability to change the user interface components according to their desires.

Usability compliance is defined as the capability of a software to adhere to standers, conventions, style guide, or usability regulations [14]. The evaluation of OSS usability compliance with ergonomic standards cannot determine the quality of the OSS, but it determines whether the OSS meets the usability requirements and recommendations that found in various ergonomic standards, laws, or/ and style guides. Among such standards are ISO 9241, ISO 14915, ISO 16071, ISO 23973, ISO/IEC 11581, ISO/IEC 18035 & 18036, ANSI CIF, and HFES/ANSI 200 [47].

Familiarity is another factor that affects the software usability [11]. According to Seffah eat al., 2006 [31] the familiarity refers to the capability of the software's interface to provide recognizable elements that can be understood by the user. In the user interface context, most of the computer users are familiar with the closed proprietary software such as Mac, Windows, and Microsoft internet explorer… etc.

Therefore, in order to improve its usability, the user-interface of Open Source Software should be designed in some way to be like components of common proprietary software. Furthermore, the familiarity could be obtained by information that provided by external resources such as advertising and word-of-mouth [32].

IV. Multistage Fuzzy Model Approach for OSS Usability

Although the literature of software usability provides many ways and methods to evaluate software usability, there is no exact approach to do so. Further, evaluation of software usability is depend on various characteristics, which are fuzzy in nature. Therefore, a new multistage fuzzy model approach for evaluating usability of OSS is presented in this paper. The advantage of using fuzzy logic is that it provides a convenient way to transform knowledge expressed in a nature language into fuzzy logic rules [31]. Multistage Fuzzy Model approach assists in decision making and control in Fuzzy-based techniques. It should be noted that, fuzzy based approaches divided into two approaches, including fuzzy traditional approach and the fuzzy reasoning approach [31]. In the next sections, a motivation of using the fuzzy reasoning approach in this paper is explained.

IV.1. Fuzzy Traditional Approach

Fuzzy traditional approach assesses software usability using all inputs (usability characteristics) by simple if-then-rules [31]. Fig. 1 presents the traditional fuzzy approach block diagram of OSS usability.

The model in Fig. 1 considers all the characteristics as inputs, In order to rank the OOS usability. This leads to generate too many rules. It is so difficult for experts to properly consider and formulate such rules [33]. Since in the simplest case, each input has three linguistic values (High, Moderate, and Low). Hence, OSS with nine usability characteristics will have a maximum number of 3^9= 19683 rules. A MATLAB-Fuzzy Tool Box could not be used to implement such fuzzy model approach.

Since a number of inputs in the MATLAB is limited to two. So the solution of this problem lies in using stage wise fuzzy reasoning approach which reduces the number of rules by dividing software into several fuzzy interface stages [6], [34] and effectively measure the usability of Open Source Software.

IV.2. Fuzzy Reasoning Approach

The proposed model is based on dividing the usability characteristics into three groups. Using stage wise fuzzy reasoning approach will also make MATLAB usage possible, since the number of inputs is limited to two in each stage. The first group includes the characteristics that reflect the capability of software to enable users to perform tasks. These characteristics are learnability, understandability, memorability, operability, attractiveness and familiarity [14], [30]. The second group comprises two characteristics, namely efficiency and error-prevention. It should be noted that, these characteristics are related to the users' capability to perform the tasks [25]-[27], [32].

Finally, the third group involves one characteristic which is usability compliance [14]. This characteristic is related to the software components. Table II shows the grouping of the grouping proposed model characteristics.

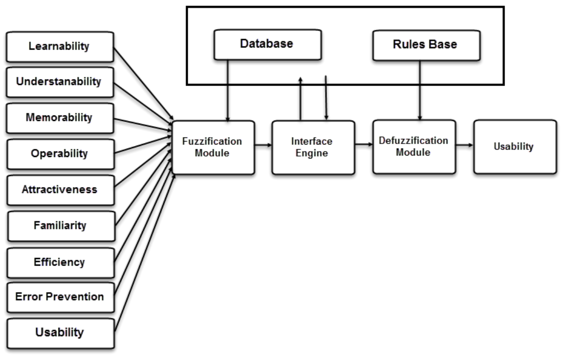


Fig. 1. Traditional fuzzy model approach of OSS usability

TABLE II
GROUPING OF CHARACTERISTICS

Group	Critical Element
Software-capability	learnability, understandability, memorability, operability, attractiveness, and familiarity
User-capability	Efficiency, and error prevention
Software-components	Usability compliance

Hence, OOS usability can be divided into multiple thought process. Learnability and understandability are used in fuzzy reasoning to determine intermediate parameter (UA1), Memorability and operability to determine another intermediate parameter (UA2), attractiveness and Familiarity to determine (UA3).

In the same way, the characteristics of the second group (efficiency and error-prevention) are used the fuzzy reasoning to determine new intermediate parameter (T) as shown in Fig. 2(a), which is later combined with the third group characteristic (usability-compliance) to determine new output parameter namely UA5.

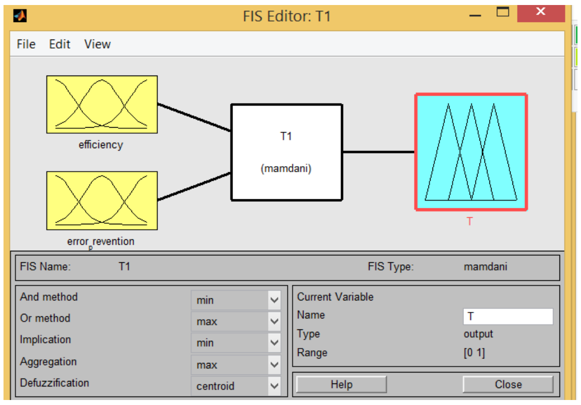
Consequently, both UA3 and AU5 are used to generate new output parameter (UAT) see Fig. 2(b).

Similarly, the output parameters UA1 and UA2 are combined to generate UA4 as new intermediate parameter which is later combined with UAT to get final output parameter (OSSU) Fig. 2(c).

IV.2.1. Fuzzification

The Fuzzification process converts crisp input values into the degree of membership using membership function.

Where, the membership function is defined as a curve that defines each point in the input space is mapped to a degree of membership of fuzzy set [35]. In the proposed model, linguistic terms have been used to represent all inputs and outputs. These linguistic terms are represented by membership functions. In this study the triangular type membership function has been used to model the membership degree of all input and output variables [37]. The range of input variables including learnability, understandability, efficiency, memorability, operability, error-prevention, attractiveness, familiarity, and usability compliance are represented as High, Moderate, and Low. In order to have a high accurate results, five membership functions Very High, High, Moderate, and Low have been used for the output parameters as shown in Fig. 3.

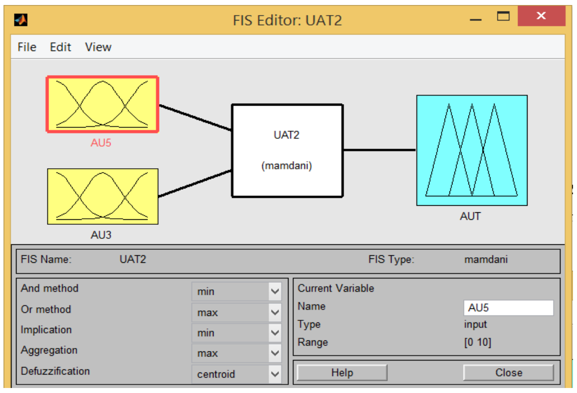


FIS Name: T1 FIS Type: mamdani

And method: min
Or method: max
Implication: min
Aggregation: max
Defuzzification: centroid

Current Variable
Name: T
Type: output
Range: [0 1]

Fig. 2(a). Learnability and usability fuzzy reasoning

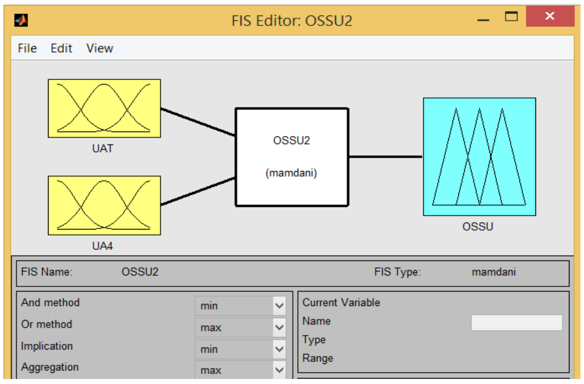


FIS Name: UAT2 FIS Type: mamdani

And method: min
Or method: max
Implication: min
Aggregation: max
Defuzzification: centroid

Current Variable
Name: AU5
Type: input
Range: [0 10]

Fig. 2(b). AU5 and AU3 fuzzy reasoning

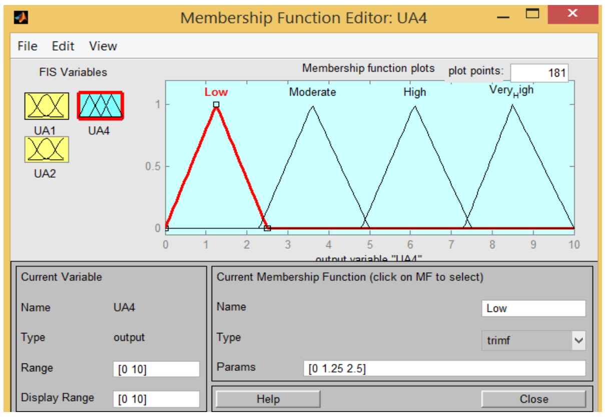


FIS Name: OSSU2 FIS Type: mamdani

And method: min
Or method: max
Implication: min
Aggregation: max

Current Variable
Name
Type
Range

Fig. 2(c). UAT and usability UA4 reasoning



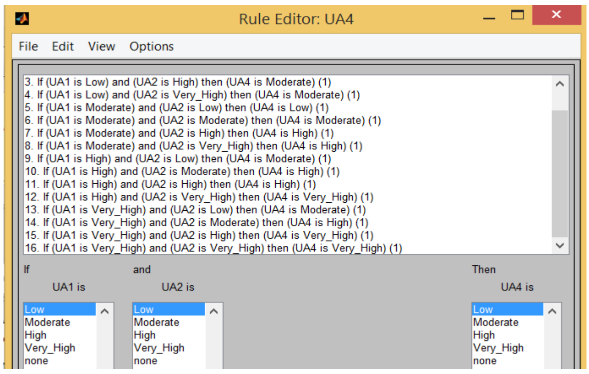
Current Variable
Name: UA4
Type: output
Range: [0 10]
Display Range: [0 10]

Current Membership Function (click on MF to select)
Name: Low
Type: trimf
Params: [0 1.25 2.5]

Fig. 3. Fuzzified output parameters

IV.2.2. Approximate Reasoning

An approximate reasoning is a process that formulated for inference to deal with ambiguous cases. It involves sub-process such as fuzzy control rules and fuzzy inference engine [31]. Fuzzy rules are linguistic IF-THEN constructions that are used to represent the human knowledge. These constructions have the general form of IF premise THEN consequence [38]. The usage of fuzzy rules exploits the tolerance of imprecision and uncertainty. This will improve the ability of human to summarize data and focus on decision-relevant information. In this study, a survey of 10 experts including academics, software developers, researcher scholars, and OSS usability experts has been done to finalize the fuzzy control rules. Fig. 4 shows the fuzzy rules of FIS1.



```
3. If (UA1 is Low) and (UA2 is High) then (UA4 is Moderate) (1)
4. If (UA1 is Low) and (UA2 is Very_High) then (UA4 is Moderate) (1)
5. If (UA1 is Moderate) and (UA2 is Low) then (UA4 is Low) (1)
6. If (UA1 is Moderate) and (UA2 is Moderate) then (UA4 is Moderate) (1)
7. If (UA1 is Moderate) and (UA2 is High) then (UA4 is High) (1)
8. If (UA1 is Moderate) and (UA2 is Very_High) then (UA4 is High) (1)
9. If (UA1 is High) and (UA2 is Low) then (UA4 is Moderate) (1)
10. If (UA1 is High) and (UA2 is Moderate) then (UA4 is High) (1)
11. If (UA1 is High) and (UA2 is High) then (UA4 is High) (1)
12. If (UA1 is High) and (UA2 is Very_High) then (UA4 is Very_High) (1)
13. If (UA1 is Very_High) and (UA2 is Low) then (UA4 is Moderate) (1)
14. If (UA1 is Very_High) and (UA2 is Moderate) then (UA4 is High) (1)
15. If (UA1 is Very_High) and (UA2 is High) then (UA4 is Very_High) (1)
16. If (UA1 is Very_High) and (UA2 is Very_High) then (UA4 is Very_High) (1)
```

Fig. 4. Fuzzy control rules of output parameters

As previously mentioned, learnability and understandability are the input parameters of first FIS and UA1 is gotten as an output parameter. Additionally, the fuzzy linguistic values of High, Moderate, and Low are assigned for the input parameters. Whereas, Very High, High, Moderate and Low are some of the output parameters. As seen from Figure above sixteen rules have been derived from the collected knowledge for the output of first FIS. E.g. if the rating for learnability is high and low for understandability then the overall rating of the output parameter will be moderate. Similarly, the rules of rest eight FISs have been framed.

V. Simulation of Multi Stage Fuzzy Approach Model for OSS Usability

The present work has been done, in order to propose a new multistage fuzzy approach model to evaluate the usability of Open Source Software. This proposed model allows to pass the fuzzy control rules to another as fact through intermediate variables (the output parameters).

In this section, the MATLAB Simulink Software is used to simulate and link all proposed model stages. In the other words, the fuzzy logic controller is used to link all FISs through passing the value of intermediate variables to another as shown in Fig. 5. To validate the proposed multistage fuzzy model approach, a simulation has been done using different sets of critical element rating see Table III. Considering the scenario in the previous table, the proposed multistage fuzzy model approach has rated the UA1, UA2, AU3, and T as 6.14, 6.14, 8.62, and 3.63 respectively. The value of T parameter has been passed and combined with the rating of usability compliance to rate the output parameter (UA5) as 3.6.

Similarly, the rating of UA1 and UA2 have been combined to rate UA4 as 6.13, using the fuzzy rules which were determined in fuzzification stage. Later, the UA3 and UA5 ratings are combined to rate UAT parameter as 6.13. Finally, the ratings of UAT and UA4 passed to the OSSU controller in order to generate the rating of overall OSS usability (6.14), then this rating has been converted to a crisp value using the MATLAB function (6).

VI. Validation of Proposed Model

A standard Analytical Hierarchy Process Technique (AHP) has been applied to validate the proposed model and to rank the usability characteristics. The AHP is a multi-criteria decision making method that was proposed by Thomas L. Saaty in 1980 [35], [36].

A survey of twenty computer science graduate students has been done to collect data, which later used with the AHP technique to evaluate the Open Source Software usability model and to rank the factors of such model.

All students have 2 to 3 years of experience using Open Source Software Linux, open office and other open source applications.

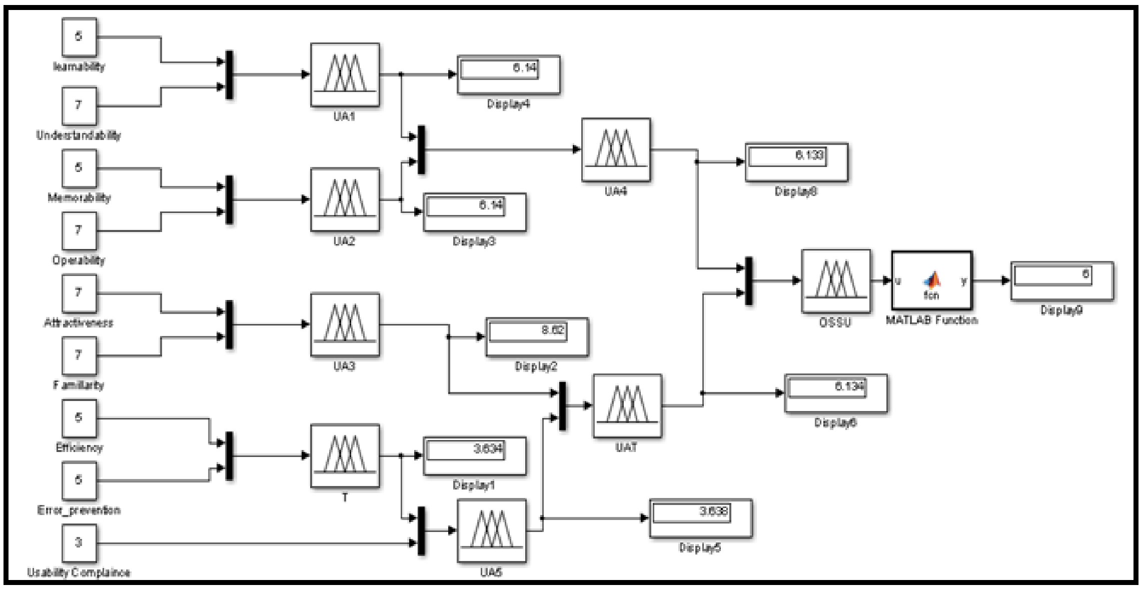


Fig. 5. Stage Wise Fuzzy Approach Model of OSS Usability-Simulink Model

TABLE III
SIMULATION DATA

Critical element	Rating	Stage wise fuzzy model output								Final Rating
		UA1	UA2	UA3	T	UA4	UA5	UAT	OSSU	
Learnability	5	6.14				6.13			6.14	6 HIGH
Understandability	7									
Memorability	5		6.14							
Operability	7									
Attractiveness	7			8.62				6.13		
Familiarity	7									
Efficiency	5				3.63		3.6			
Error-prevention	5									
Usability compliance	3									

TABLE IV
CHARACTERISTICS VALUES USING AHP TECHNIQUE

	L	U	EF	ME	ER	O	A	UC	F	9th	W	A.W	Lamda
L	1.00	1.66	3.43	4.02	3.98	5.00	5.80	6.00	4.57	3.47	0.2729	2.70	9.88
U	0.60	1.00	2.51	3.97	3.87	5.10	6.60	6.50	4.50	3.05	0.2402	2.30	9.58
EF	0.29	0.40	1.00	3.28	3.00	4.53	5.60	5.70	3.82	2.05	0.1612	1.58	9.83
ME	0.25	0.25	0.30	1.00	0.73	3.80	5.00	5.30	1.83	1.11	0.0874	0.86	9.84
ER	0.25	0.26	0.33	1.37	1.00	3.91	4.72	5.00	1.38	1.16	0.0913	0.89	9.80
O	0.20	0.20	0.22	0.26	0.26	1.00	4.06	5.00	1.54	0.64	0.0504	0.52	10.38
A	0.17	0.15	0.18	0.20	0.21	0.25	1.00	4.50	0.58	0.37	0.0291	0.31	10.54
UC	0.17	0.15	0.18	0.19	0.20	0.20	0.22	1.00	0.47	0.25	0.0195	0.20	10.46
F	0.22	0.22	0.26	0.55	0.72	0.65	1.72	2.12	1.00	0.61	0.0481	0.44	9.17
									Total	12.70	1.0000	9.81	9.94

The survey consists of 36 comparison pair wise questions to compare nine usability characteristics with each other, in order to determine that which characteristic is more important and gets more priority for Open Source Software.

The mean of each question's responses to the pairwise weight values of the main usability characteristics are filled in a square matrix Table IV.

After getting the pairwise relative weight values of the usability characteristics, the nth root has been calculated by multiplying all row values and then taking the $(1/9)^{th}$ root of each usability characteristic [24].

As also mentioned in Table IV the nth values of the usability characteristics model are, for learnability (3.47), for understandability (3.05), for efficiency (2.05), memorability (1.11), for error-prevention (1.16), for operability (0.64), for attractiveness (0.37), for usability compliance (0.25), and for familiarity (0.61). The sum of these values is 12.7. Next Eigen Vector (ω) has been calculated by dividing the nth root value of each characteristic by the sum of all nth root values of the characteristics [39]. It could be seen that, the eigenvector of relative importance or in the other words the weights of the main usability characteristics of Open Source Software from the students' perspective as the following: learnability (0.27), understandability (0.24), efficiency (0.16), error prevention (0.091), memorability (0.087), operability (0.05), familiarity (0.048), attractiveness (0.029), and usability compliance (0.019).It is worth to mention that, the summation of eigenvector values is 1.0, hence the comparison values for the usability characteristics are right [19].

Thus, from students' perspectives learnability of the OSS is the most important one over other usability characteristics, followed by understandability, efficiency, error-prevention, memorability, portability, familiarity, attractiveness, and finally the usability compliance is less important [40]. These results provide more than ranking. In fact, the relative weights are a ratio scale could be divided among such usability characteristics.

Human judgment is not always consistent, thus in order to check the consistency of participants' answers, the Consistency Index (CI) and Consistency Ratio (CR) must be obtained, Eqs. (2) and (3).

To do so the eigenvalue is required, Eq. (1) [39]:

$$A = \lambda max\ \omega, \text{ and } \lambda max > n \tag{1}$$

where λ eigenvalue; and n is represents the number of elements to be compared. The difference between λ and n is an indication of judgment consistency:

$$CI = \frac{\lambda\, max - n}{n-1} \tag{2}$$

$$CR = \frac{CI}{RI} \tag{3}$$

where CR is the average of consistency index and should satisfy the condition of CR <= 0.1.

For this study, all the values of the eigenvalue have been mentioned in the Table IV. These values reveal that λ value has satisfied the condition of $\lambda > n$, (9.94 > 9).

Consequently, by applying Eqs. (2) and (3), CI= (9.94 – 9) / 8 = 0.1175 and CR= 0.1175/1.45= 0.081, is being content with the condition of CR < 0.1 [39].

VII. Contribution to the Study

Fuzzy logic allows infinite intermediate logical values between the false and true. Another study in this regard represents mapping of analytical tools for modeling and computational treatment of Fuzzy Systems.

This mapping showed the existence of relationships and the appropriateness of using the Fuzzy Theory in building models for solving problems involving dynamical systems and, in particular, of Chaotic Dynamical Systems [33]. It was proposed to differentiate between two types of Dynamic Fuzzy Systems that is Intrinsic Fuzzy Dynamical Systems and Dynamical Systems Fuzzy Extrinsic. Dynamic systems were used in Fuzzy modeling for trajectories of fixing problems in chaotic systems.

Case studies were developed that allowed to verify through simulations and tests in billiards Chua circuit (the latter implemented in physical prototype) the appropriateness of using this technique in solving these problems. Similarly, the qualitative research presented here aims to discuss the value and role of fuzzy logic in solving real problems, given the characteristics of its tools to deal with subjective questions, since the troubleshooting from the real world are loaded relationships built in the inner world of the resolver are from the subjectivity of the subject resolver [41].

Further studies conducted to determine one of the goals of modern-day open system and to answer the questions: What are the theoretical assumptions of fuzzy theory and what possibilities recognition of their value and their role in software development education? Software development and relative reliability linked with fuzzy reasoning.

Without a doubt, Fuzzy control systems have been widely used in process control industry. Normally controlling analog variables, such as pressure, temperature, flow, position and velocity. However, the use of this topology in the Information System and Software Development field is relatively new.

VIII. Conclusion

This paper proposed a new multistage fuzzy model approach for Open Source Software usability. This model enhances the usability of Open Source Software by adapting and quantifying nine usability characteristics in the Open Source Software context. These characteristics include learnability, understandability, efficiency, error-prevention, memorability, operability, attractiveness, and usability-compliance. Based on the fuzzy control rules that generated from experts' knowledge, and by using the MATLAB Software, the proposed model evaluates and quantifies the usability of OSS.

In order to evaluate the validity of the proposed usability model and ranking its characteristics, Analytic Hierarchy Process (AHP) has been applied. Pairwise relative weights of characteristics have been taken through a survey of twenty graduate students.

The results have revealed that the proposed model is vailed for this context and sequence the usability characteristics of this model according to their importance as follows: learnability, understandability, efficiency, error prevention, memorability, operability, familiarity, attractiveness, and finally usability-compliance [42]. Hence, these results should be taken into account as usability requirements when open source applications being designing and implemented.

Acknowledgements

The authors of this research work are highly grateful to each and every individual who facilitated and supported in the completion of this study.

The research process required a higher level of commitment and dedicated efforts, and the facilitators were a source of motivation and enrichment in this study.

References

[1] B. Ashley, & K. Hyunju, A survey on open source software licenses, *Journal of Computing Sciences in Colleges*, *Vo,l. 22,* n. 4, pp. 252-259, 2007.

[2] S. M. A. Shah, G. I. G. Al-Matroushi, M . F. Qureshi, Usability Assessment of Open source application, *International Journal of Advanced Research in Computer Science*, *Vol. 4*, n. 2, 2013.

[3] M. F. Qureshi, & G. I. G. Al-Matroushi, Usability Assessment of Open source application. *International Journal of Advanced Research in Computer Science*, *Vol. 4*, n. 1, pp. 13-19 , 2013.

[4] A. Madan, & S. K. Dubey, Usability evaluation methods: a literature review, *International Journal of Engineering Science and Technology*, *Vol. 4, n.* 2, pp. 590-599, 2012.

[5] J. Hall, *Usability Themes in Open Source Software* Ph.D. Thesis, Dept. Computer Science, University of Minnesota, Minnesota, USA, 2014.

[6] A. Shaout, M. Al-Shammari, Fuzzy logic modeling for performance appraisal systems: a framework for empirical evaluation, *Expert systems with Applications*, Vol. *14*, n.3, pp. 323-328, 1998.

[7] A. Oztekin, Z. J. Kong, & O. Uysal, UseLearn: A novel checklist and usability evaluation method for eLearning systems by criticality metric analysis, *International Journal of Industrial Ergonomics*, Vol. *40*, n. 4, pp. 455-469, 2010.

[8] A. Raza, L. F. Capretz, & F. Ahmed, An open source usability maturity model (OS-UMM). *Computers in Human Behavior*, *28*(4), pp. 1109-1121, 2012.

[9] J. Horsky, K. McColgan, J. E. Pang, A. J. Melnikas, J. A. Linder, J. L. Schnipper, B. Middleton, Complementary methods of system usability evaluation: surveys and observations during software design and development cycles, *Journal of biomedical informatics, Vol. 43*, n. 5, pp. 782-790, 2010.

[10] A. Fernandez, S. Abrahao, & E. Insfran, Empirical validation of a usability inspection method for model-driven Web development, *Journal of Systems and Software*, Vol. *86*, n. 1, pp. 161-186, 2013.

[11] A. Karahoca, E. Bayraktar, E. Tatoglu, & D. Karahoca, Information system design for a hospital emergency department: A usability analysis of software prototypes, *Journal of biomedical informatics*, *Vol. 43*, n. 2, pp. 224-232, 2010.

[12] R. R. Bond, D. D. Finlay, C. D. Nugent, G. Moore, & D. Guldenring, A usability evaluation of medical software at an expert conference setting, *Computer methods and programs in biomedicine*, Vol. *113,* n. 1, pp.383-395, 2014.

[13] S. K. Dubey, & A. Rana, Analytical comparison of usability measurement methods, *International Journal of Computer Applications*, Vol. *39*, n. 15, 2012.

[14] I. S. O. (2001). IEC 9126-1: Software Engineering-Product Quality-Part 1: Quality Model. *Geneva, Switzerland: International Organization for Standardization*..

[15] P. Rogers Sharp, *Interaction Design-beyond human-computer interaction* (John Wiley & Sons 2015).

[16] A. McCall, P. K. Richards & G. F. Walters, Factors in Software Quality, US Rome Air Development Center Reports, US Department of Commerce, Washington DC, 1977.

[17] K. D. Eason, Towards the experimental study of usability, *Behaviour and Information Technology*, Vol. 3, n.2, pp. 133-143, 1984.

[18] B. Shackel, Usability—Context, framework, definition, design and evaluation, in B. Shackel and S. Richardson, Human Factors for Informatics Usability, Cambridge, MA: University Press, pp. 21–38, 1991.

[19] J. Nielsen, Usability Engineering (Elsevier 1993)

[20] B. Fitzgerald, The transformation of Open Source Software, MIS Quarterly, Vol. 30, n. 3, pp. 587–598, 2006.

[21] R. E. Al-Qutaish, Quality models in software engineering literature: an analytical and comparative study, *Journal of American Science*, Vol. *6,* n.3, pp. 166-175, 2010

[22] G. S. Laterza. Valenti, A. Cucchiarelli, & M. Panti, Computer based assessment
systems evaluation via the ISO9126 quality model, *Journal of Inf ormation Technology Education, Vol. 1*, N.3, pp. 157-175, 2002.

[23] T. A. Kroeger, N. J., Davidson, & S. C. Cook, Understanding the characteristics of quality for software engineering processes: A Grounded Theory investigation. *Information and Software Technology*, Vol. *56*, n. 2, pp. 252-271, 2014.

[24] T. A. Al-Rawashdeh, F. M. Al'azzeh, S. M. Al-Qatawneh, Evaluation of ERP Systems Quality Model Using Analytic Hierarchy Process (AHP) Technique. *Journal of Software Engineering and Applications*, *Vol. 7*, n. 4, pp. 225-232, 2014.

[25] A. Kumar, P. S. Grover, & R. Kumar, A quantitative evaluation of

aspect-oriented software quality model (AOSQUAMO), *ACM SIGSOFT Software Engineering Notes*, Vol. *34*, n. 5, pp. 1-9, 2009.

[26] P. Booth, *An introduction to human-computer interaction. Hillsdale*, (Lawrence Erlbaur Associates Publishers, 1989).

[27] Benson, M. Muller-Prove, & J. Mzourek, *Professional usability in open source projects: GNOME, OpenOffice. org, NetBeans*, CHI'04 extended abstracts on Human Factors in Computing Systems (page:1083 year of publication 2004).

[28] S. L. Pauwels, C. Hubscher, S. Leuthold, J. A. Bargas-Avila, & K. Opwis, Error prevention in online forms: Use color instead of asterisks to mark required-fields, *Interacting with Computers*, *Vol. 21*, n. 4, pp. 257-262, 2009.

[29] N. Bevan, *ISO and industry standards for user centered design*. Retrieved November 23 (2000), 2010.

[30] A. Maedche, A. Botzenhardt, & L. Neer, *Software for people: Fundamentals, trends and best practices* (Springer Science & Business Media, 2012).

[31] A. Seffah, M. Donyaee, R. B. Kline, H. K. Padda, Usability measurement and metrics: A consolidated model, *Software Quality Journal*, Vol. *14, n.* 2, pp. 159-178, 2006.

[32] L. Casalo, C. Flavián, & M. Guinaliu, The role of perceived usability, reputation, satisfaction and consumer familiarity on the website loyalty formation process, *Computers in Human Behavior*, Vol. *24*, n. 2, pp. 325-345, 2008.

[33] Y. Shin, L. Williams, Can traditional fault prediction models be used for vulnerability prediction, *Empirical Software Engineering*, Vol. *18*, n. 1, pp. 25-59, 2013.

[34] A. Shaout, J. Trivedi, Performance appraisal system using a multistage fuzzy architecture, *International Journal of Computer and Information Technology, Vol. 2,* n. 3, pp. 405-411, 2013.

[35] T. L. Saaty, *What is the analytic hierarchy process?* (Springer Berlin Heidelberg 1988).

[36] T. J. Ross, *Fuzzy logic with engineering applications* (John Wiley & Sons. 2009).

[37] K. Dahal, Z. Hussain, & M. A. Hossain, *Loan risk analyzer based on fuzzy logic. Proceedings*. The IEEE International Conference on e-Technology, e-Commerce and e-Service (page: 363 year of publication 2005).

[38] S. K. Dubey, & A. Rana, Fuzzy Model for Quantifying Usability of Object Oriented Software System, *International Journal of Computer Science and Information Security*, Vol. *10*, n. 4 2012.

[39] L. A. Zadeh, Fuzzy sets. *Information and control*, Vol. *8*, n. 3, pp. 338-353 , 1965.

[40] E. M. Fredericks, B. DeVries, & B. H. Cheng, AutoRELAX: automatically RELAXing a goal model to address uncertainty, *Empirical Software Engineering*, *Vol. 19*, n. 5, pp. 1466-1501, 2014**.**

[41] B. Adams, R. Kavanagh, E. A. Hassan, & D. M. German, An empirical study of integration activities in distributions of Open Source Software.*Empirical Software Engineering, Vol. 20*, n. 1, pp. 1-42, 2015.

[42] S. M. Alnaeli, J. I. Maletic, & M. L. Collard, An empirical examination of the prevalence of inhibitors to the parallelizability of Open Source Software systems. *Empirical Software Engineering, Vol. 20*, n.2 pp. 1-30, 2015.

[43] B. W. Boehm, J. R. Brown, H. Kaspar, H. Lipow, G. McLeod, & M. Merritt*, Characteristics of Software Quality*, (North Holland Publishing, 1978).

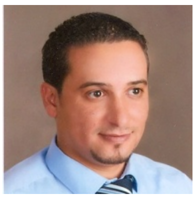
[44] R. G. Dromey, Cornering the chimera. *IEEE Software*, *Vol. 1*, n. 1, pp. 33-43, 1996.

[45] S. Fahmy, N. Haslinda, W. Roslina, & Z. Fariha, Evaluating the Quality of Software in e-Book Using the ISO 9126 Model, *International Journal of Control and Automation*, *Vol. 5*, n. 2, pp. 115-122, 2012.

[46] ISO 9241, *Ergonomics requirements for office work with visual display terminals (VDTs) – Part 11: Guidance on usability*, 1998.

Authors' information

Department of Computer Science, Alzaytoonah University of Jordan.
E-mail: thamer.r@zuj.edu.jo



Dr. **Thamer A. Al-Rawashdeh** received his doctorate in software engineering from Utara University of Malaysia. Before coming to Alzaytoonah University, he was instructor at Utara University of Malaysia for the following courses: managerial mathematic, Information Systems, VB.NET. Dr. Alrawashdeh is assistant professor at Alzaytoonah University of Jordan, where he has taught Object-Oriented Analysis and Design, Object-Oriented Programming (JAVA, C++, and VB.NET), Computer Networks, Information System, ASP.NET, System Analysis and Design, Project Management and Software Quality Assurance. His research interest is in the software testing, software development, software quality, and software usability. In addition to teaching and researching, he is chair of computer science department at Alzatoonah University of Jordan.