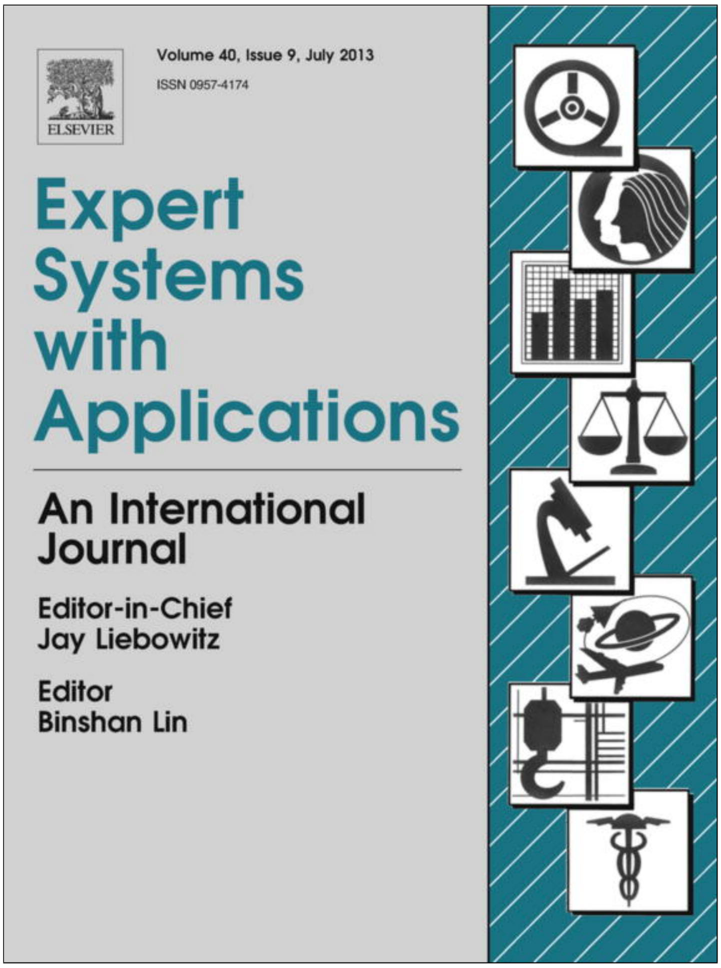

Volume 40, Issue 9, July 2013
ISSN 0957-4174
ELSEVIER
Expert Systems with Applications
An International Journal
Editor-in-Chief
Jay Liebowitz
Editor
Binshan Lin

Expert Systems with Applications 40 (2013) 3313–3327

Contents lists available at SciVerse ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Neuro-Fuzzy Expert System for evaluating the performance of Distributed Software System Architecture

B.A. Akinnuwesi [a,*], Faith-Michael E. Uzoka [b], Abayomi O. Osamiluyi [c]

[a] Department of Information Technology, Bells University of Technology, Ota, Ogun State, Nigeria
[b] Department of Computer Science & Information Systems, Mount Royal University, Calgary, Canada
[c] Department of Computer Science and Technology, Bells University of Technology, Ota, Ogun State, Nigeria

ARTICLE INFO

Keywords:
Distributed Software System Architecture
Evaluation
Responsiveness
Performance
Expert system
Organizational variable

ABSTRACT

A Neuro-Fuzzy Performance Evaluation Model (NFPEM) proposed in Akinnuwesi, Uzoka, Olabiyisi, and Omidiora (2012) was reviewed in this work with the view of modifying it and thus making it flexible and scalable. The neuro-fuzzy expert system (NFES) reported in this paper is an enhancement to NFPEM with expert system components. NFES can be used to evaluate the performance of Distributed Software System Architecture (DSSA) with user-centric variables as parameters for performance measurement. The algorithm developed for NFES was implemented using Coldfusion programming language and MySQL relational database management system. The prototype of NFES was simulated using some life data and the performance results obtained point to the DSSA responsiveness to the users' requirements that are defined at the requirements definition phase of the software development process. Thus the performance value is a qualitative value representing DSSA (i.e. system) responsiveness.

1. Introduction

A Distributed Software System (DSS) is a complex system used by organizations to deploy services simultaneously to many people online and in real time (Akinnuwesi, 2011). The decisions made at each phase of the DSS development process impact on the quality attributes (e.g. reusability, reliability, modifiability and performance) of software (Lloyd & Connie, 1998). Performance is a pertinent quality attribute of software systems. It is an indicator of the extent to which software system/components meet the requirements of the end users. Performance failure usually results in damaged customer relations, loss of revenue, loss of productivity and cost overruns due to tuning or redesign of system (Lloyd & Connie, 1998). Therefore it becomes imperative to analyze and predict the expected performance of DSS at the architectural design level in order to: avoid the pitfalls of poor quality of software at system implementation level; provide all organizational services and also satisfy the performance expectations of the stakeholders such that all stakeholders get maximum satisfaction from the software system.

A survey of DSS performance evaluation models was carried out in (Olabiyisi, Omidiora, Uzoka, Victor, & Akinnuwesi, 2010), Olabiyisi et al. (2011) and Akinnuwesi et al. (2012) considering performance of the system at both architectural and implementation levels. The authors deduced that none of the existing models considered evaluating DSSA performance using contextual organizational variables and this informed the development of NFPEM that was presented in (Akinnuwesi, 2011 and Akinnuwesi et al., 2012). NFPEM is a neuro-fuzzy algorithm used to measure performance of DSSA based on 31 contextual organizational variables. Though the model did the evaluation as required but in the course of reviewing it, the following were identified as needed to enhance the functionality of the model: (1) Incorporation of components of expert system; (2) Inclusion of values (i.e. input, intermediate results and final output values) reusable components; (3) Making the model dynamic and scalable such that a performance engineer can define the input contextual variables peculiar to each organization as well as the machine function.

Vlahavas, Stamelos, Refanidis, and Tsoukias (1999) used Multiple Criteria Decision Aid (MCDA) framework to build an expert system for software evaluation. The authors evaluated software performance at the implementation level which is not very ideal because evaluation at the architectural level helps to establish the adequacy of the architecture in meeting organizational requirements before developing it to a complete software system (Samuel, 2006; Samuel & Alejandro, 2003; Simonetta, Roberto, & Moreno, 2004). This helps to minimize the risk of tuning and redesigning the system if it has performance failure at the point of implementation by users.

* Corresponding author.
E-mail addresses: boluakinnuwesi@bellsuniversity.edu.ng, moboluwaji@gmail.com (B.A. Akinnuwesi).

0957-4174/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.eswa.2012.12.039

In this paper, NFES is developed by adopting expert system principle to modify NFPEM algorithm in order to address the aforementioned limitations in both (Akinnuwesi et al., 2012 and Vlahavas et al., 1999). The rest of the paper is organized as follows: Review of related literature is presented in Section 2. The conceptual design of NFES and its algorithm are presented in Section 3. NFES implementation is presented in Section 4. Some conclusions are drawn in Section 5.

2. Literature review

In this research we carried out a detail review of Vlahavas et al. (1999), Behrouz, Vani, and Abdel-Halim (2009) and Akinnuwesi et al. (2012) in order to establish the system components needed to enhance NFPEM functionalities.

2.1. Vlahavas et al. (1999)

The authors presented an expert system for performance evaluation based on the Multiple Criteria Decision Aid (MCDA) framework with features of flexibility in problem modeling and built-in knowledge about software problem and software attribute assessment. This means that different kinds of problems can be modeled using the system, thus, the type of problem and the attributes to be considered for performance measurement are selected or determined using the expert assistant component. Fig. 1 presents the architecture of Vlahavas et al. (1999).

Performance measurement attributes were broken down into basic and compound attributes. The compound attributes were broken down into sub-attributes while the basic ones cannot be further analyzed. Focus was on quality and cost attributes. The quality attribute being a compound one was further broken down into functionality, reliability, usability, efficiency, maintainability, and portability attributes; each of which is further analyzed into sub-attributes until basic ones are reached and cannot be further analyzed. Scale of measurement is associated with each basic attribute and at the point of evaluation, a value is provided for each attribute of each software product to be analyzed as specified by the software performance engineer. Results of performance evaluation were stored for future reference by the evaluator and determination of attributes for evaluation of other products at a later time. Profiles of its users were maintained. All these were stored in the knowledge base.

The system becomes difficult or complex to use when attributes become much (i.e. >150). It requires the user of the system to have at least software engineering or performance evaluation skills. All sub-attributes that make up a compound attribute may not be determined and redundancy of basic attributes may even occur. Also, it is not subjective enough because attribute values are determined by the evaluator rather than by a collection of opinions from the different users of the product. The primary aim of software performance evaluation is to determine if requirements specification defined by the client organization (end users) is met. The approach defined in Vlahavas et al. (1999) investigates performance at end users implementation level of the software product and not at the architectural level where performance evaluation is considered very necessary.

2.2. Behrouz et al. (2009)

The authors proposed a system similar to that of Vlahavas et al. (1999) but it is based on the Multidimensional Weighted Attribute Framework (MWAF). They assigned weights and values to attributes and applied the principles of Tukey's pairwise comparison and analysis of variance (ANOVA) tests to them. The systems presented in Behrouz et al. (2009) and Vlahavas et al. (1999) both have the same limitations. Behrouz et al's architecture is presented in Fig. 2.

2.3. Akinnuwesi et al. (2012)

The need for evaluating performance of DSSA using user-centric variables was established in Olabiyisi et al. (2010) and Olabiyisi et al. (2011) and thus NFPEM was proposed in Akinnuwesi et al.

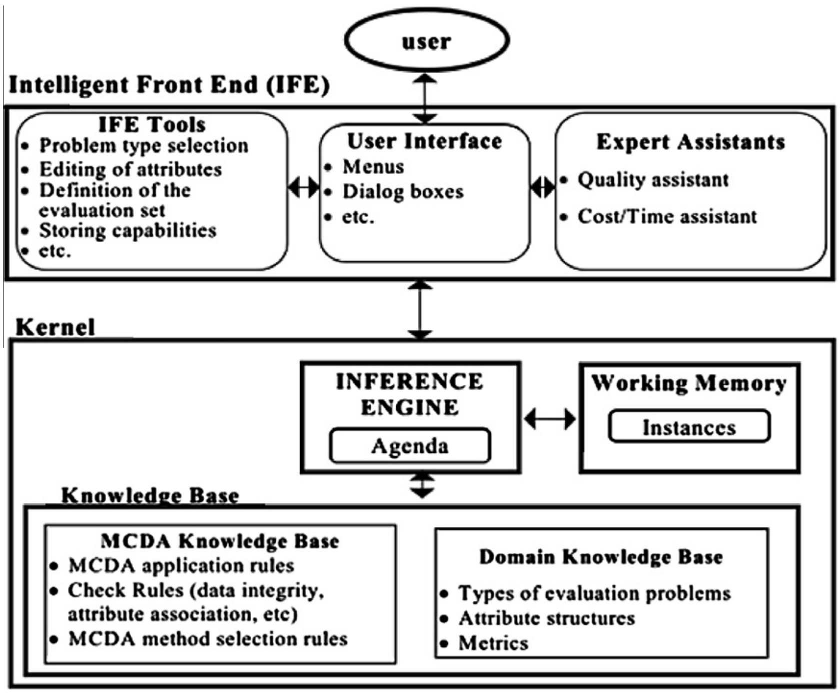


Fig. 1. The structure of ESSE (Vlahavas et al., 1999).

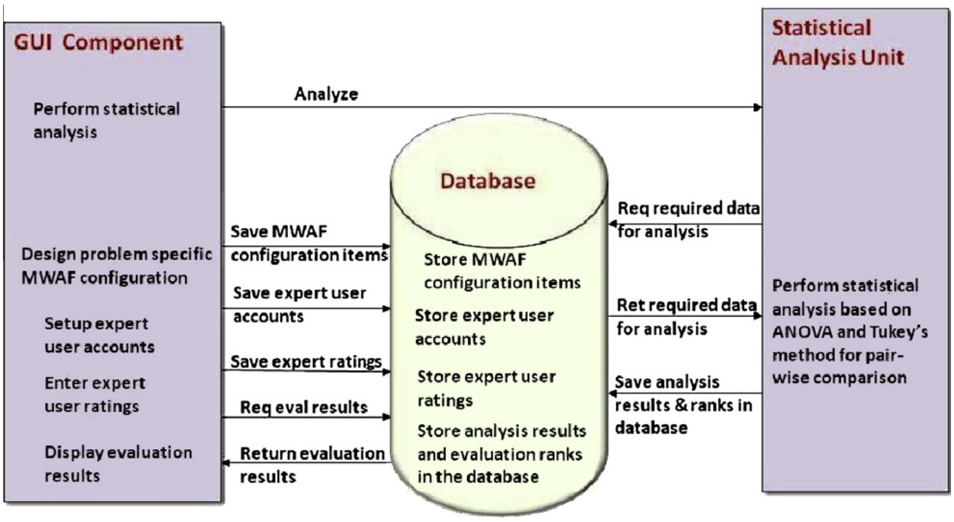


Fig. 2. GPSE system component interaction (Behrouz et al., 2009).

(2012). NFPEM is a neuro-fuzzy hybridized model that is used to evaluate the performance of software architecture using contextual organizational (i.e. user-centric) parameters for evaluation. NFPEM is composed of 31 contextual organizational variables ($x_{i,i=1,2,3,..31}$), 10 software constructs ($y_{j,j=1,2,3,...10}$) and a matching function that comprises of 10 linear regression functions that maps the contextual variables with the respective software construct. The contextual organizational variables are the functional and non-functional requirements of the organization that are incorporated into the software architecture and their impact depends on elements such as people, structure, technology and the external environment where the organization operates. NFPEM architecture is presented in Fig. 3.

The contextual variables are: x_1 = Communication rules with external organizations (CRE1), x_2 = Data communication rules and semantics within the client organization (DCRO), x_3 = Willingness of users for IT training (WUIT), x_4 = IT infrastructure available in client organization (ITIF), x_5 = Budget of the client organization for software project (BSPJ), x_6 = Feasibility study done by the project team in client organization (FSTU), x_7 = Expected size of the organization database (SODB), x_8 = Policies for interoperability (PIN1), x_9 = Defined mapping of data with external business entity and services (DMEB), x_{10} = Users definition for input data and the format for input (UDI1), x_{11} = Data input validation strategy/procedure defined by client organization (DVSC), x_{12} = Developers' understanding of the organization's goal and task (DUOG), x_{13} = Internal services of

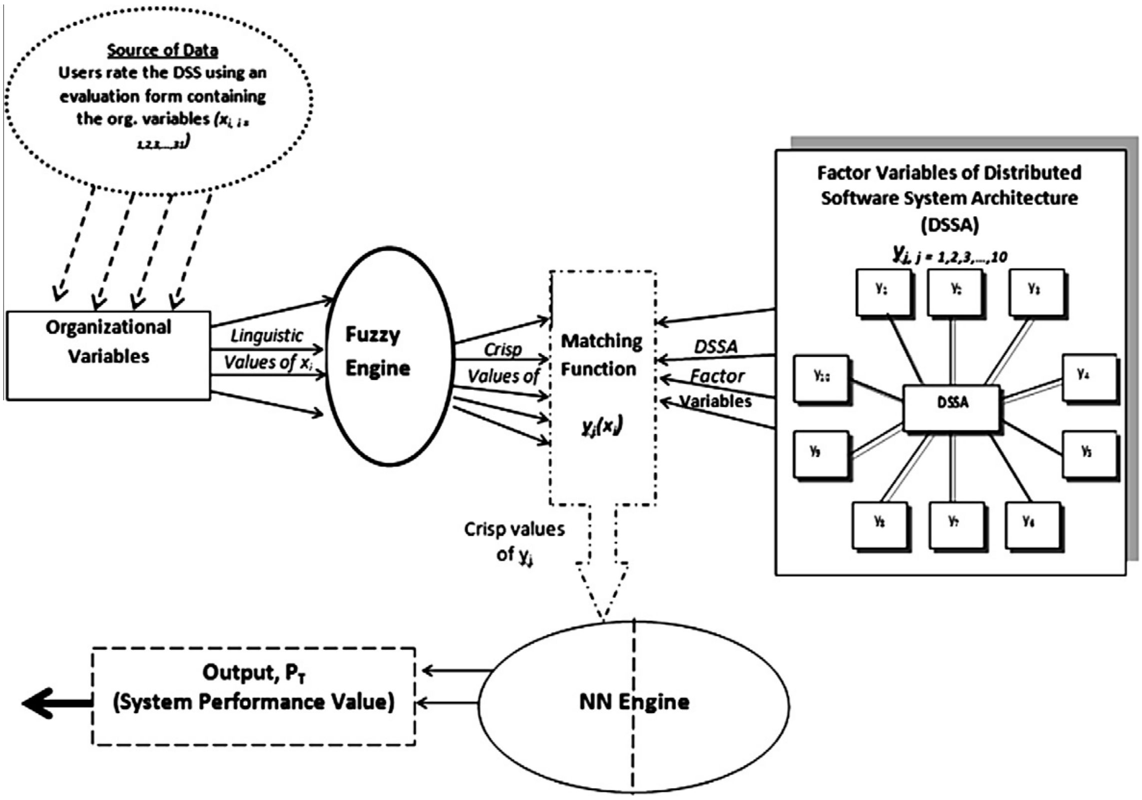


Fig. 3. Neuro-Fuzzy based user-centric Performance Evaluation Model (NFPEM) (Akinnuwesi et al., 2012).

the client organization and their relationships (ISO1), x_{14} = Professional qualification of users (PQUS), x_{15} = Academic qualification of users (AQUS), x_{17} = Involvement of users in system design (USDE), x_{18} = Involvement of users in system operation (USOP), x_{19} = Population of users expected to use/operate the system (PUOS), x_{21} = Information requirements of users and the format in which it expected (UIRF), x_{22} = Organization goals and tasks (OGTS), x_{23} = Organization policies/procedure for transaction flow (OPTF), x_{24} = Organization defined functions required in the user interface (ODFI), x_{25} = Organization defined access right for users of applications (DUAR), x_{26} = Business rules associated with the data to be processed (BRDP), x_{27} = Data security measures put in place by the organization (ODS1), x_{22} = Organizations goals and tasks (OGTS), x_{28} = Data flow procedure (DFP1), x_{29} = Defined timeout for services/operations (DTSO), x_{30} = External services requested by the client organization from external organizations (ESEO), x_{31} = Message contract for communication between organizations (MCC1).

The software constructs are: y_1 = Business entity, y_2 = Preparedness of the client organization, y_3 = Service agent, y_4 = Process and presentation logic, y_5 = Users interest and IT expertise, y_6 = User involvement, y_7 = User interface, y_8 = Data access and security, y_9 = Business workflow, y_{10} = Service layer;

The subjective views of end users are obtained using software performance assessment form that is filled by each end user. The linguistic values chosen for rating each performance variable are: *"Strongly Satisfied"*, *"Satisfied"*, *"Fairly Satisfied"*, *"Dissatisfied"* and *"Strongly Dissatisfied"*. The users also state his/her confidence level for each variable that is rated. The confidence level scale is in the range 1 (lowest) to 10 (highest).

The authors gave a detailed description of the structure and algorithm of NFPEM in the paper.

The following were observed as limitations: (1) The system lacks knowledge processing ability, which is provided by expert system components; (2) The model accept values into the evaluation parameters, processes the values and produce output without storing any of the input, intermediate values generated and the output for future purposes (i.e. values are not reusable); (3) The model does not have features for flexibility and scalability because its input variables (i.e. 31 user-centric variables) and machine function (that is comprised of ten different equations) are fixed.

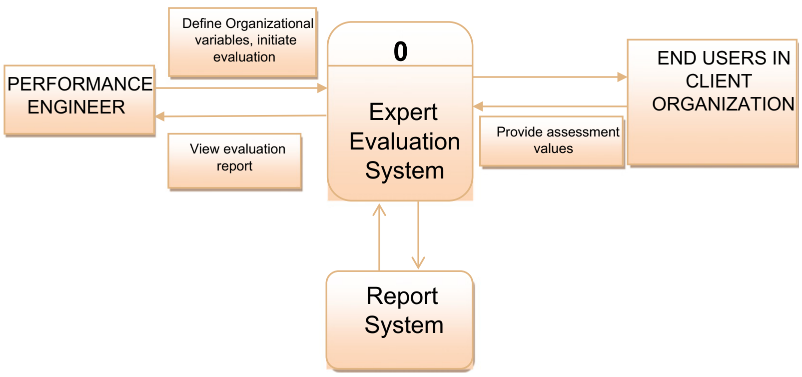


Fig. 4. Context diagram of NFES to evaluate performance of Software System Architecture.

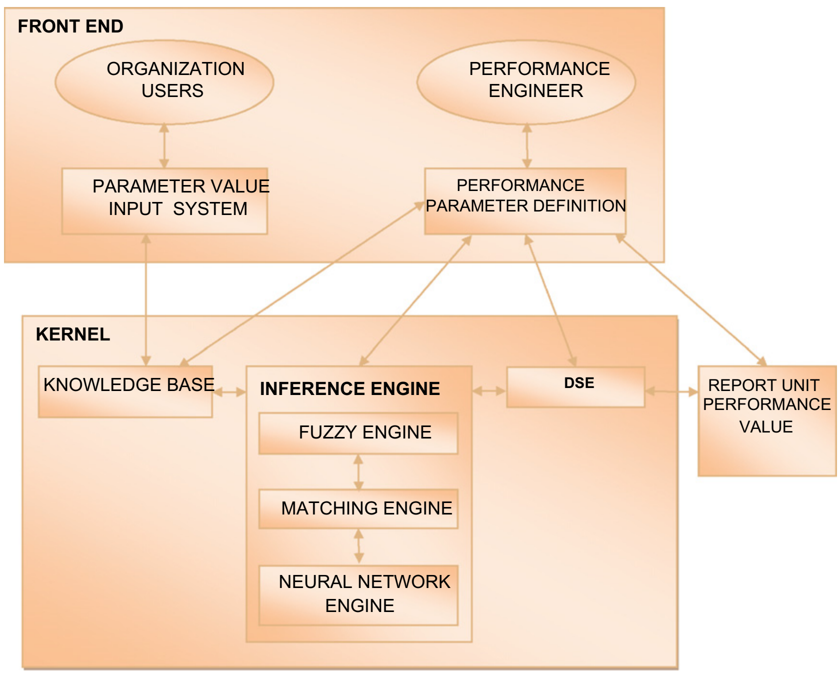
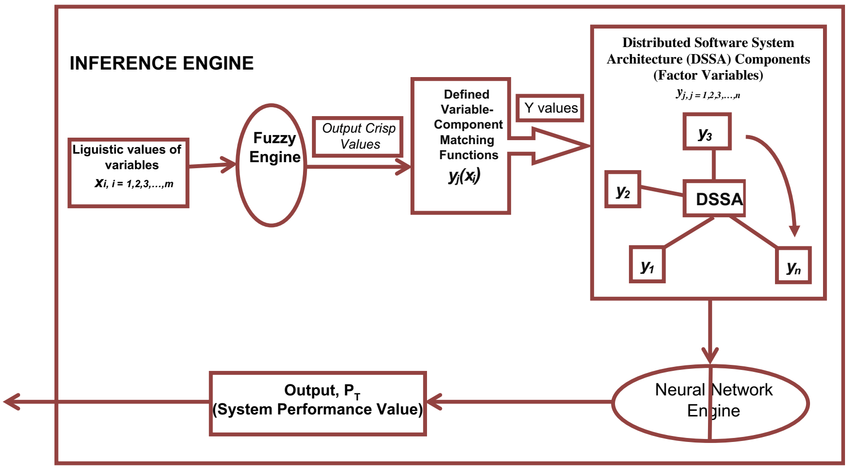


Fig. 5. Architecture of NFES for evaluating performance of Distributed Software System Architecture.



where $x_{i,\ i=1,2,3,\dots,m}$ and $y_{j,\ j=1,2,3,\dots,n}$ are varying number of variables.

Fig. 6. Modified inference engine structure showing relationship among fuzzy, matching and neural network engines.

The limitations observed in the performance evaluation models reported in the above literatures motivated the development of NFES that is reported in this paper.

3. Proposed Neuro-Fuzzy Expert System (NFES) for evaluating DSSA

The proposed expert system is an enhanced system over the works of Akinnuwesi (2011), Akinnuwesi et al. (2012) and Vlahavas et al. (1999).

3.1. NFES architecture

The architecture of NFES is made up of the following components:

a. Domain expert (performance engineer)
b. System end users (staff and customers of client organization)
c. Knowledgebase
d. Inference Engine
   i. Fuzzy engine
   ii. Matching engine
   iii. Neural network engine
e. Decision Support Engine (DSE)
f. Report Engine

Table 1
Performance values (linguistic labels and values).

Linguistic Labels	Very poor	Poor	Good	Very good	Excellent
Values	0.00–0.99	1.00–1.99	2.00–2.99	3.00–3.99	4.0–5.00

The context diagram of the architecture is presented in Fig. 4. It gives an overview of NFES. The architecture of NFES is presented in Fig. 5. The identified users include the performance engineer and the end users of the client organization.

The ***Performance Engineer*** refers to the domain expert that defines the organizational variables to be used during evaluation of the software system architecture. The ***end users*** in the client organization provide the linguistic values as response to the linguistic variables in the performance assessment form which can either be completed online or offline. There is provision for the definition of the organizational variables to suit any given client organization. This is because all client organizations cannot be assumed to be affected by the same organizational factors. This makes the variable definition unit of NFES to be flexible and scalable compared to that of NFPEM which is fixed to 31 contextual organization variables. The data provided by the ***end users*** are stored in the ***Knowledgebase*** for use during performance evaluation. The ***knowledgebase*** stores both temporary and permanent data. Temporary data include those stored for use during evaluation process and is later

Table 2
Database relations.

Relation name	Description
COMPANYINFO	This relation stores data about the client organization whose distributed software architecture is being evaluated
USERS	This relation stores data about the end-users of the client organization are enrolled in the system for evaluation
ENGINEERS	This relation stores data about performance engineer. It includes their contact information and login details
OPTION_TYPES	This relation stores the different option types that may be used in a performance assessment form. Examples include the YES, NO and Strongly Agree, Agree, Not Sure, Disagree, Strongly Disagree option types
OPTIONS	This relation stores data about the different options under a particular option type
QUESTIONNAIRES	This relation stores defined performance assessment forms created by performance engineer
VARIABLES	This relation stores the different variables (questions) attached to a particular questionnaire
NEURALALGORITHMS	This relation stores the different types of neural network algorithm supported by the neural network engine
PROBLEMS	This relation stores the definition the different evaluation scenarios referred to as problems. A performance assessment form is attached to the problem
Y_VALUES	This relation stores the different components of software architecture and the mathematical relationship between the component and the variables defined in the performance assessment form as identified by the performance engineer
RESPONSES	This relation stores the various responses of organizational users to the performance assessment form variables and their rating confidence
CRISPS	This relation stores the crisp value of responses to each variable specified in a particular problem's performance assessment form

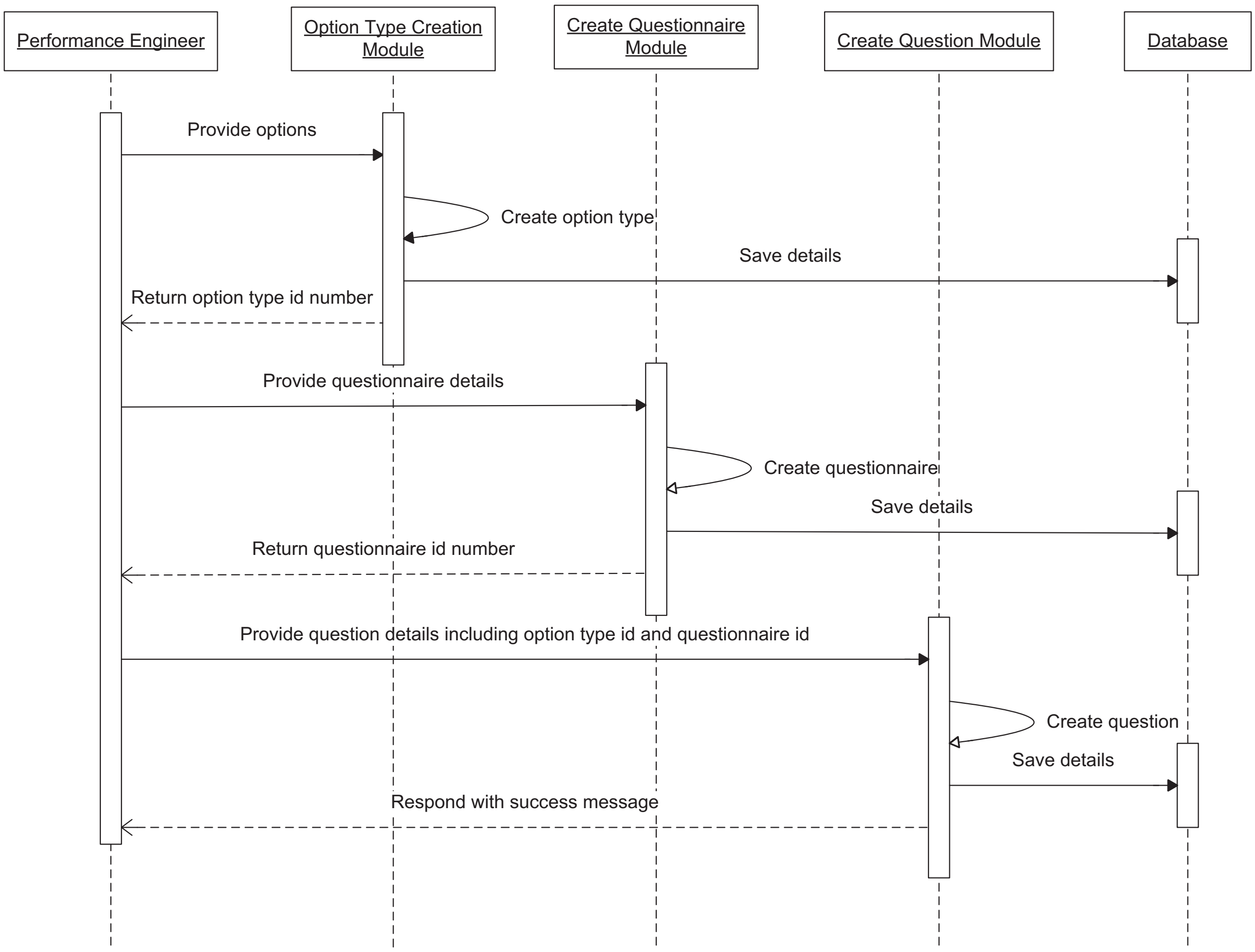


Fig. 7. Create assessment form sequence diagram.

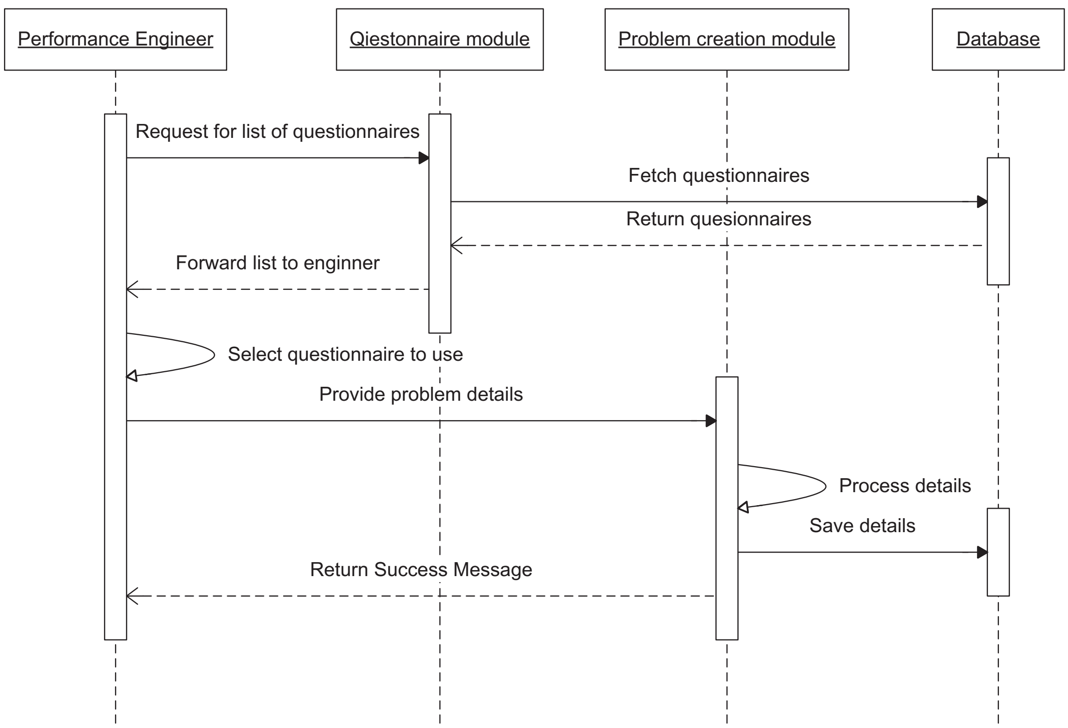


Fig. 8. Define problem sequence diagram.

cleared. Permanent data include those that remain permanent in the system. They include data such as the defined organizational variables which are used by the system. After obtaining the assessment views of the end users, the performance engineer initiates the evaluation process. This involves the ***Inference Engine*** retrieving the data from the knowledgebase and using them for computation of performance value. Also, intermediate values are stored in the knowledgebase for reuse (both temporary and permanent data). The inference engine comprises of the fuzzy engine, matching engine and neural network engine as sub-components that all work together to achieve the purpose of processing responses from the organizational users into a reasonable value with which decision can be made by the ***Decision Support Engine***. The structure of the ***Inference Engine*** is presented in Fig. 6. The sequence diagrams for some of the operations are presented in Figs. 7–10.

The algorithm used in the ***Inference Engine*** is a modified version of NFPEM algorithm presented in Akinnuwesi (2011) and Akinnuwesi et al. (2012). The matching function of NFES is flexible and scalable. It is a set of regression equations mapping a set of organizational variables with given software construct. The process of establishing required organizational variables to be used for evaluation and also loading the variables on respective software architecture construct has been presented in Akinnuwesi (2011) and Akinnuwesi et al. (2012). The modified algorithm for the inference engine of NFES is described below:

3.1.1. Modified Fuzzification and Defuzzification Logic Algorithm

Step 1: Input values for $\mathbf{x_{ij}}$, $i = 1,2,3,\ldots,g$ and rating confidence $\mathbf{C_{ij}}$ Where $j = 1,2,3,\ldots,k$ (k = total number users sampled to collect data for $\mathbf{x_{ij}}$) and $i = 1,2,3,\ldots,g$ (g = total number of variables) *Note:* Values for $\mathbf{x_{ij}}$ and $\mathbf{C_{ij}}$ are obtained from users of DSS via the DSSA performance assessment form stored in the expert system

Step 2: Compute normalized rating confidence of users, $\boldsymbol{\alpha}_{ij}$, using KAM normalization procedure described in Akinnuwesi (2011) and Akinnuwesi et al. (2012)

Step 3: Adjust rated values of users for each jth variable using

$$\varphi_{\mathbf{i,j}=}\boldsymbol{\alpha}_{\mathbf{i,j}}\{\mathbf{u}_{\mathbf{t}-1,}\mathbf{u}_{\mathbf{t}},\mathbf{u}_{\mathbf{t}+1}\}$$

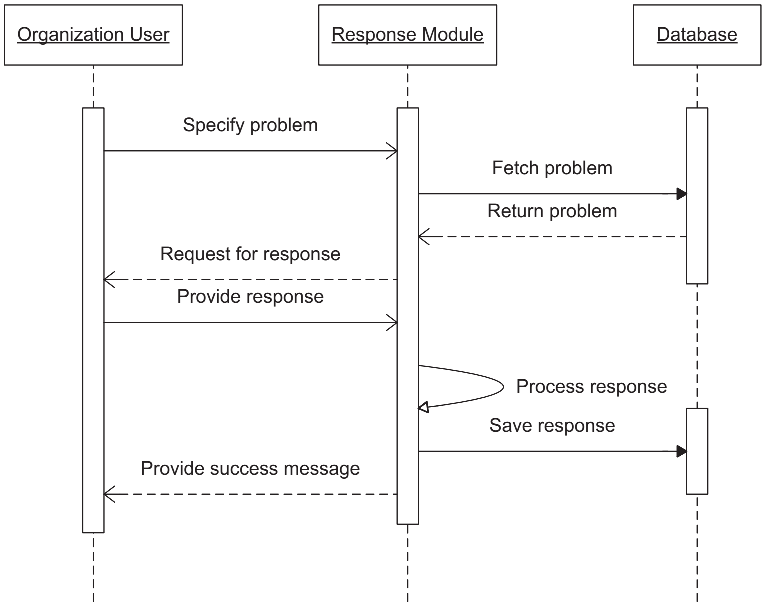


Fig. 9. Respond to performance assessment form sequence diagram.

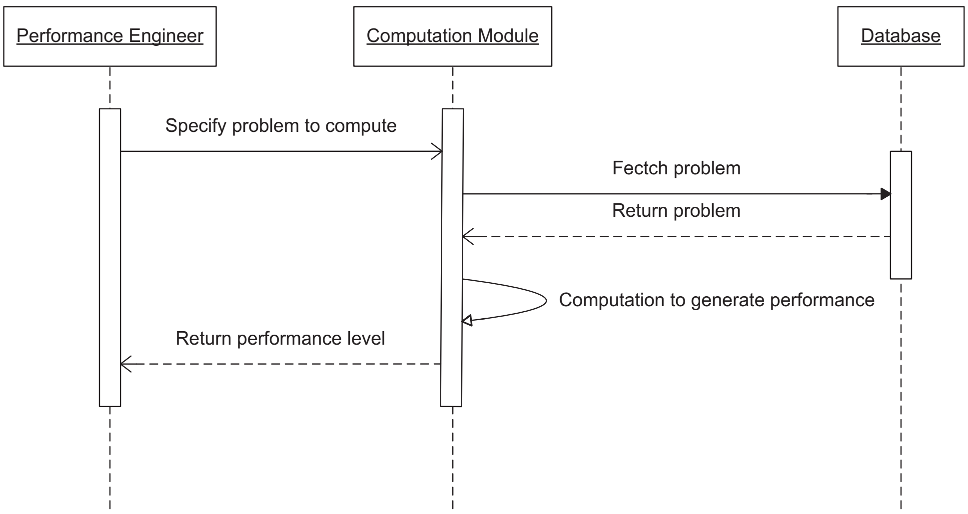


Fig. 10. Compute performance sequence diagram.

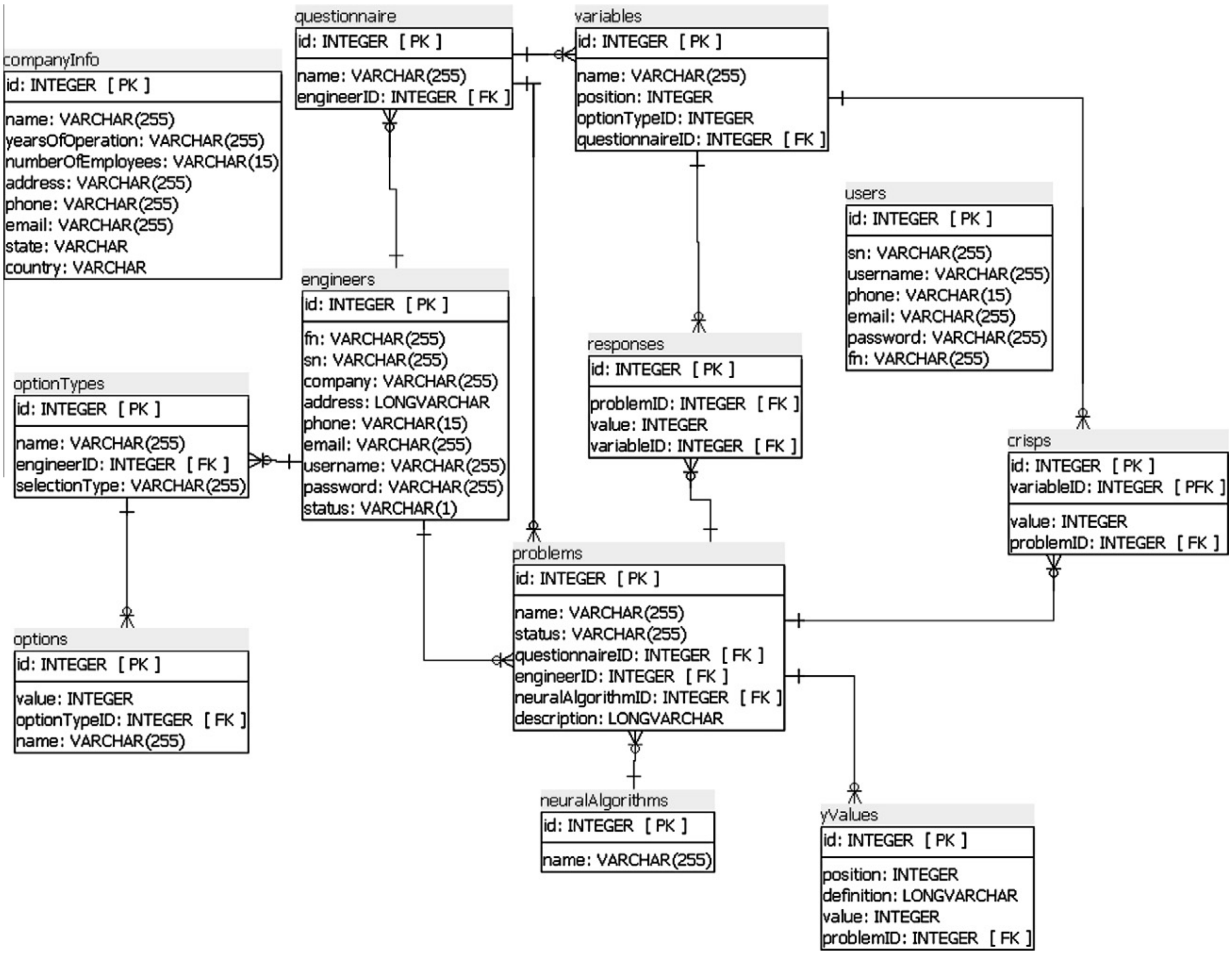


Fig. 11. Entity-relationship diagram for the database of NFES.

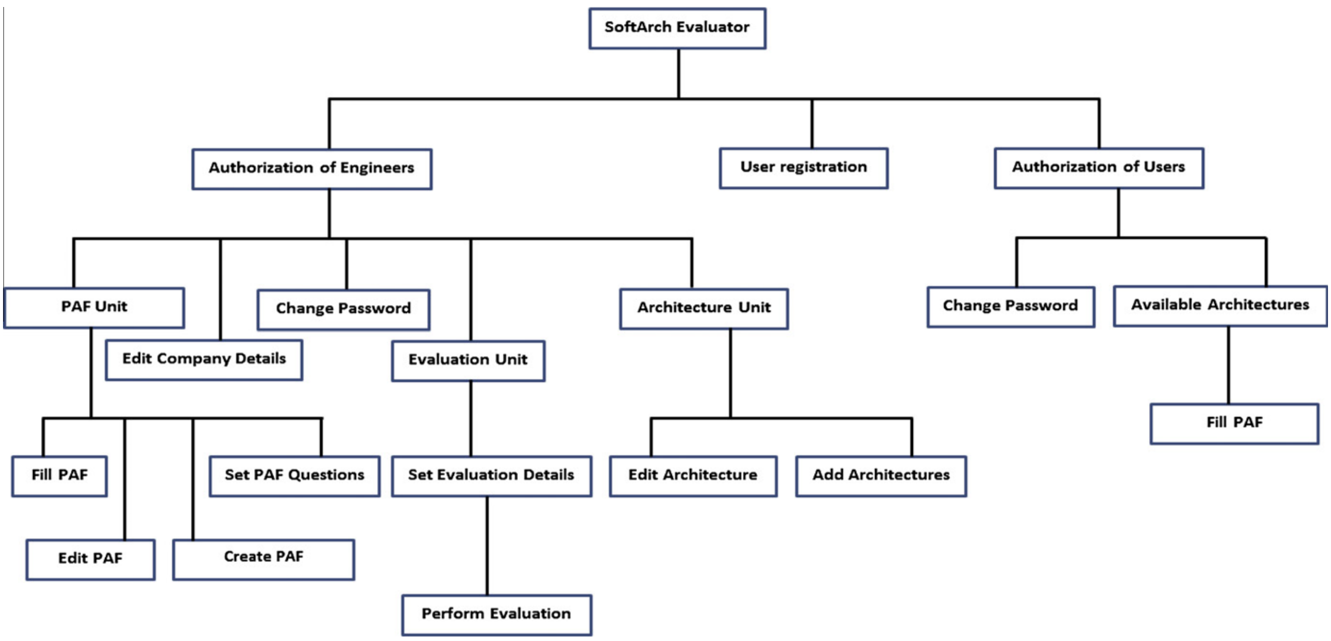


Fig. 12. NFES implementation hierarchy.

Step 4: Compute the membership values of the adjusted rated values, $\boldsymbol{\varphi}_{i,j}$, of users, using membership functions defined in Akinnuwesi (2011) and Akinnuwesi et al. (2012)

Step 5: Compute the crisp value of $\mu_X\{\boldsymbol{\varphi}_{i,j}\}$ using the defuzzification function,

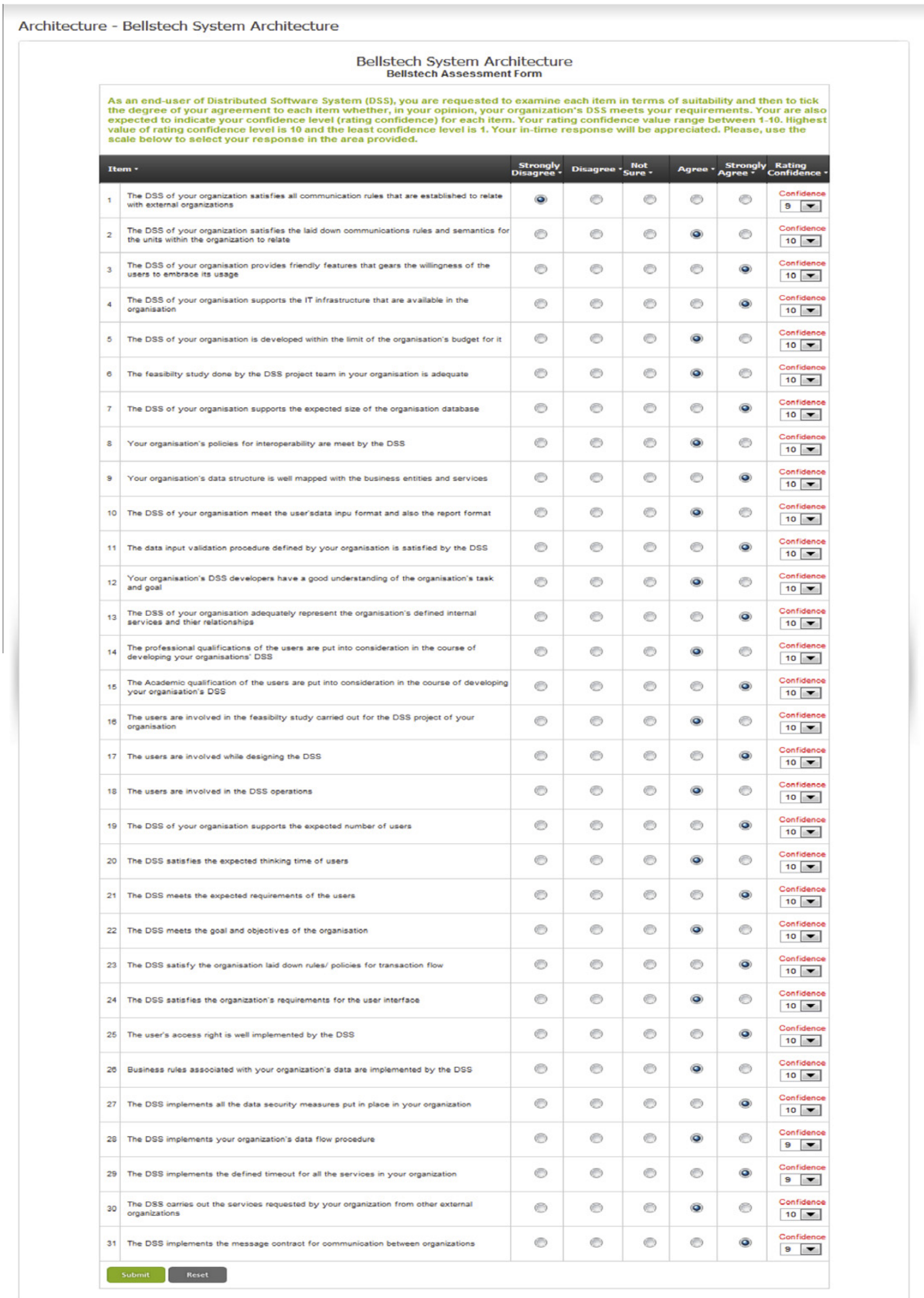
$$\hat{\mathbf{z}}_{i,j} = \frac{\sum \boldsymbol{\varphi}_{i,j}(\mu_x(\boldsymbol{\varphi}_{i,j}))}{\sum \mu_x(\boldsymbol{\varphi}_{i,j})}$$

where: $\hat{z}_{i,j}$= Crisp value obtained; μ_X $(\phi_{i,j})$ = Fuzzy membership values

Step 6: Compute the mean xi of $\hat{z}_{i,j}$, $i = 1,2,3,\ldots,k$ and $j = 1,2,3,\ldots,g$; using

$$\mathbf{x}_i = \frac{\sum_{j=1}^{n} \hat{z}_{i,j}}{n}$$

Step 7: Compute values of $\mathbf{y_j}$, $j = 1,2,3,\ldots,n$, using the defined matching function for the system in question. where: $\mathbf{y_j}$



Architecture - Bellstech System Architecture

Bellstech System Architecture
Bellstech Assessment Form

As an end-user of Distributed Software System (DSS), you are requested to examine each item in terms of suitability and then to tick the degree of your agreement to each item whether, in your opinion, your organization's DSS meets your requirements. Your are also expected to indicate your confidence level (rating confidence) for each item. Your rating confidence value range between 1-10. Highest value of rating confidence level is 10 and the least confidence level is 1. Your in-time response will be appreciated. Please, use the scale below to select your response in the area provided.

	Item	Strongly Disagree	Disagree	Not Sure	Agree	Strongly Agree	Rating Confidence
1	The DSS of your organization satisfies all communication rules that are established to relate with external organizations	●	○	○	○	○	Confidence 9
2	The DSS of your organization satisfies the laid down communications rules and semantics for the units within the organization to relate	○	○	○	●	○	Confidence 10
3	The DSS of your organisation provides friendly features that gears the willingness of the users to embrace its usage	○	○	○	○	●	Confidence 10
4	The DSS of your organisation supports the IT infrastructure that are available in the organisation	○	○	○	○	●	Confidence 10
5	The DSS of your organisation is developed within the limit of the organisation's budget for it	○	○	○	●	○	Confidence 10
6	The feasibilty study done by the DSS project team in your organisation is adequate	○	○	○	●	○	Confidence 10
7	The DSS of your organisation supports the expected size of the organisation database	○	○	○	○	●	Confidence 10
8	Your organisation's policies for interoperability are meet by the DSS	○	○	○	●	○	Confidence 10
9	Your organisation's data structure is well mapped with the business entities and services	○	○	○	○	●	Confidence 10
10	The DSS of your organisation meet the user'sdata inpu format and also the report format	○	○	○	●	○	Confidence 10
11	The data input validation procedure defined by your organisation is satisfied by the DSS	○	○	○	○	●	Confidence 10
12	Your organisation's DSS developers have a good understanding of the organisation's task and goal	○	○	○	●	○	Confidence 10
13	The DSS of your organisation adequately represent the organisation's defined internal services and thier relationships	○	○	○	○	●	Confidence 10
14	The professional qualifications of the users are put into consideration in the course of developing your organisations' DSS	○	○	○	●	○	Confidence 10
15	The Academic qualification of the users are put into consideration in the course of developing your organisation's DSS	○	○	○	○	●	Confidence 10
16	The users are involved in the feasibilty study carried out for the DSS project of your organisation	○	○	○	●	○	Confidence 10
17	The users are involved while designing the DSS	○	○	○	○	●	Confidence 10
18	The users are involved in the DSS operations	○	○	○	●	○	Confidence 10
19	The DSS of your organisation supports the expected number of users	○	○	○	○	●	Confidence 10
20	The DSS satisfies the expected thinking time of users	○	○	○	●	○	Confidence 10
21	The DSS meets the expected requirements of the users	○	○	○	○	●	Confidence 10
22	The DSS meets the goal and objectives of the organisation	○	○	○	●	○	Confidence 10
23	The DSS satisfy the organisation laid down rules/ policies for transaction flow	○	○	○	○	●	Confidence 10
24	The DSS satisfies the organization's requirements for the user interface	○	○	○	●	○	Confidence 10
25	The user's access right is well implemented by the DSS	○	○	○	○	●	Confidence 10
26	Business rules associated with your organization's data are implemented by the DSS	○	○	○	●	○	Confidence 10
27	The DSS implements all the data security measures put in place in your organization	○	○	○	○	●	Confidence 10
28	The DSS implements your organization's data flow procedure	○	○	○	●	○	Confidence 9
29	The DSS implements the defined timeout for all the services in your organization	○	○	○	○	●	Confidence 9
30	The DSS carries out the services requested by your organization from other external organizations	○	○	○	●	○	Confidence 10
31	The DSS implements the message contract for communication between organizations	○	○	○	○	●	Confidence 9

Submit Reset

Fig. 13. On-line performance assessment form fill page.

is the jth software construct/component specified and n = total number of architecture constructs/components specified.

Step 8: Neural Network process starts Invoke the NN algorithm: $\boldsymbol{NN(y_j)}$ [$\boldsymbol{j = 1\ldots n}$]

Step 9: Algorithm terminates.

3.1.2. Neural Network algorithm

Step 1: Assign constant values: $\boldsymbol{\eta}$ (NN learning rate), where $0 < \eta \leqslant 1$; $\mathbf{Q}$ (define d threshold Performance value), where $0.0 \leqslant \mathbf{Q} \leqslant 1.0$ and whether to use the threshold value or not. Initialize $\mathbf{w_i}$ (multiplicative weight), $0.0 \leqslant \mathbf{w_j} \leqslant 1.0$, $j = 1,2,3,\ldots,n$ (n = number of architecture components specified)

Step 2: Execute the summation function: $P = \sum w_j y_j$; $j = (1,2,\ldots,n)$ and $\mathbf{y_j}$ is gotten from the fuzzy algorithm

Step 3: Execute the normalization function: If using threshold

$$f(P) = P_T = \left\{ \begin{array}{ll} P & if\,(0.0 \leqslant P \leqslant 1.0) \text{ and } (P \geqslant Q) \\ \frac{1}{1+e^{-P}} & if\ P < 0, P > 1.0 \end{array} \right\}$$

Else (i.e. not using threshold value)

$$f(P) = P_T = \left\{ \begin{array}{ll} P & if\ (0.0 \leqslant P \leqslant 1.0) \text{ and } (P \geqslant P_{r-1}) \\ \frac{1}{1+e^{-P}} & if\ P < 0, P > 1.0 \end{array} \right\}$$

where r is the rth iteration. If r is 1, then $\mathbf{P_{r-1} = 0}$ Either ways, if $\boldsymbol{P_T} = \mathbf{P}$ then output $\mathbf{P}$ and Go to **Step 5**; otherwise Go to **Step 4**

Step 4: Delta training Compute delta, $\boldsymbol{\delta}$: $\boldsymbol{\delta = Q - P}$ Adjust weights $\mathbf{w_j}$ using delta weight adjustment function: $\mathbf{w_j = w_j + \eta\delta y_j}$, $j = 1,2,\ldots,n$ Go to **Step 3**.

Step 5: Step 5: Algorithm terminates.

The ***Decision Support Engine (DSE)*** takes the output from the ***Inference Engine*** and produces a meaningful output for the performance engineer based on fuzzy and neural network functions defined in the algorithm. The output of the ***Inference Engine*** is crisp and falls in the range, 0.00–5.00. The crisp value is converted to a linguistic value using Table 1. The ***Report Engine*** is responsible for presenting all the reports envisaged for the system as required by the performance engineer. (See Table 2).

3.2. Knowledgebase design

This section presents the different entities represented in the knowledgebase of the system and the entities–relationships (E–R).

The knowledgebase includes the rule-base that is coded into the system and the database which is designed using MySQL database management system. The entities and their relationship are represented using both rule-base and database. The rule-base is composed of the fuzzy rules and the neural network processing rules defined in NFES algorithm. The database consists of data about the performance engineer, client organization, performance evaluation problem, organizational users, performance assessment forms created by the performance engineer and responses to the performance assessment forms as provided by the organizational users. Relational data model is adopted to represent the entities in the database for the system. The relations are semantically related and this is subject to the relationships among the entities.

The general form of a relation is as follows (Cannolly & Begg, 2002; Kroenke, 1992):

$$R[A_1, A_2, \ldots A_k, A_{k+1}, \ldots A_{n-1}, A_n]$$

The name of the relation is represented by R, the set $\{A_i\}$, $i = 1,2,\ldots,n$, represents the attributes of the relation R. The following relations are considered in the database:

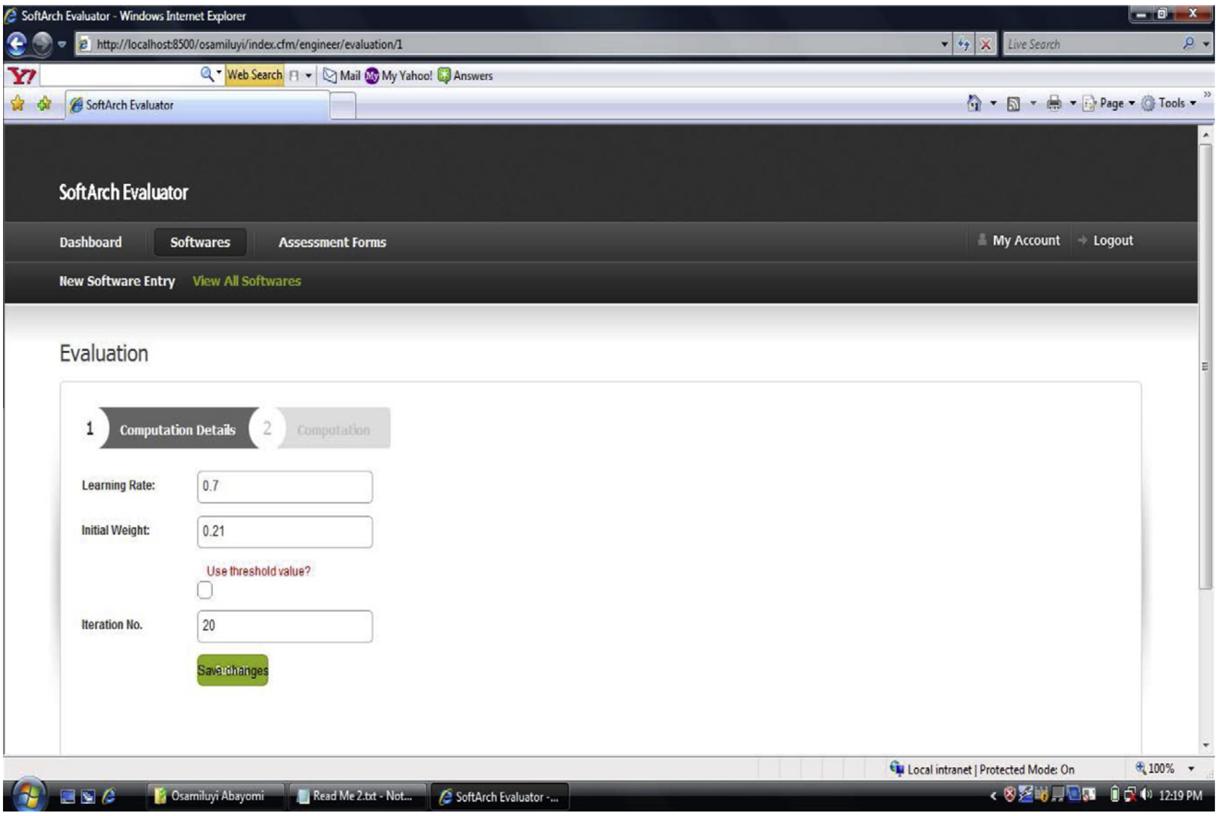


Fig. 14. Interface to input simulation parameters.

The schema for the database is as follows:

COMPANYINFO [id, name, yearsOfOperatioin, numberOfEmployees, address, phone, email, state, country]
QUESTIONNAIRE [id, name, engineerID]
VARIABLES [id, name, position, optionTypeID, questionnaireID]
ENGINEERS [id, fn, sn, company, address, phone, email, username, password, status]
OPTION_TYPES [id, name, engineerID, selectionType]
OPTIONS [id, value, optiontypeID, name]
RESPONSES [id, problemID, value, variableID]
PROBLEMS [id, name, status, questionnaireID, engineerID, neuralAlgorithmID, description]
NEURALALGORITHMS [id, name]
Y_VALUES [id, position, definition, value, problemID]
CRISPS [id, variableID, value, problemID]
USERS [id, sn, user name, phone, email, password,fn]

Fig. 11 presents the E–R diagram. This shows the logical relationships of the data objects of the entities.

4. System implementation

This section discusses the implementation of NFES vis-à-vis the functions, features and composition of both the front-end and the back-end. The implementation tools used are: Coldfusion, MySQL and NAVICAT database management applications.

4.1. NFES implementation hierarchy

NFES implementation hierarchy is presented in Fig. 12. It is designed for two categories of users: the organization users and performance engineer. The hierarchy shows the various operations that can be performed with the system by the users. Thus the system consists of the performance engineer module which comprises of several other sub modules that are integrated to achieve the functions of a performance engineer (i.e. upload client details, create performance assessment questions, create performance assessment form, upload details of client's system architecture, evaluate performance of client's system architecture etc.). NFES also consists of the organizational user module which allows each organizational user to view bio-data, view

Table 3
Simulation result.

Neural network simulation parameters	
Organization name	Bells university of technology
Learning rate	0.7
Initial weight	0.21
Number of iteration	20
Total organizational users involved in the evaluation	20

Neural network simulation values

Iteration	Synaptic weights	P_{n-1}	P_n	Error $(P_n - P_{n-1})$	Performance output %
1	w[1] = 0.210000, w[2] = 0.210000, w[3] = 0.210000, w[4] = 0.210000, w[5] = 0.210000, w[6] = 0.210000, w[7] = 0.210000, w[8] = 0.210000, w[9] = 0.210000, w[10] = 0.210000	0	0.0676	0.0676	6.76
2	w[1] = 0.140656, w[2] = 0.111813, w[3] = 0.302615, w[4] = 0.095947, w[5] = 0.150526, w[6] = 0.223757, w[7] = 0.081144, w[8] = 0.133227, w[9] = 0.117649, w[10] = 0.151469	0.0676	0.2581	−0.1906	25.81
3	w[1] = 0.336197, w[2] = 0.388687, w[3] = 0.041453, w[4] = 0.417562, w[5] = 0.318236, w[6] = 0.184964, w[7] = 0.444501, w[8] = 0.349717, w[9] = 0.378067, w[10] = 0.316519	0.2581	0.0042	0.254	0.42
4	w[1] = 0.075575, w[2] = 0.019662, w[3] = 0.389537, w[4] = −0.011096, w[5] = 0.094707, w[6] = 0.236668, w[7] = −0.039791, w[8] = 0.061173, w[9] = 0.030975, w[10] = 0.096535	0.0042	0.4168	−0.4126	41.68
5	w[1] = 0.498956, w[2] = 0.619144, w[3] = −0.175926, w[4] = 0.685260, w[5] = 0.457829, w[6] = 0.152675, w[7] = 0.746942, w[8] = 0.529913, w[9] = 0.594826, w[10] = 0.453899	0.4168	0.0001	0.4166	0.01
6	w[1] = 0.071420, w[2] = 0.013779, w[3] = 0.395086, w[4] = −0.017929, w[5] = 0.091144, w[6] = 0.237493, w[7] = −0.047511, w[8] = 0.056574, w[9] = 0.025442, w[10] = 0.093029	0.0001	0.5107	−0.5106	51.07
7	w[1] = 0.595404, w[2] = 0.755709, w[3] = −0.304741, w[4] = 0.843893, w[5] = 0.540550, w[6] = 0.133540, w[7] = 0.926163, w[8] = 0.636694, w[9] = 0.723274, w[10] = 0.535308	0.5107	0	0.5107	0
8	w[1] = 0.071325, w[2] = 0.013644, w[3] = 0.395213, w[4] = −0.018086, w[5] = 0.091062, w[6] = 0.237512, w[7] = −0.047688, w[8] = 0.056468, w[9] = 0.025315, w[10] = 0.092948	0	0.5129	−0.5129	51.29
9	w[1] = 0.597622, w[2] = 0.758850, w[3] = −0.307704, w[4] = 0.847541, w[5] = 0.542452, w[6] = 0.133100, w[7] = 0.930284, w[8] = 0.639149, w[9] = 0.726228, w[10] = 0.537180	0.5129	0	0.5129	0
10	w[1] = 0.071324, w[2] = 0.013643, w[3] = 0.395214, w[4] = −0.018087, w[5] = 0.091062, w[6] = 0.237512, w[7] = −0.047690, w[8] = 0.056467, w[9] = 0.025314, w[10] = 0.092948	0	0.5129	−0.5129	51.29
11	w[1] = 0.597636, w[2] = 0.758869, w[3] = -0.307722, w[4] = 0.847563, w[5] = 0.542464, w[6] = 0.133098, w[7] = 0.930310, w[8] = 0.639165, w[9] = 0.726246, w[10] = 0.537192	0.5129	0	0.5129	0
12	w[1] = 0.071324, w[2] = 0.013643, w[3] = 0.395214, w[4] = −0.018087, w[5] = 0.091062, w[6] = 0.237512, w[7] = −0.047690, w[8] = 0.056467, w[9] = 0.025314, w[10] = 0.092948	0	0.5129	−0.5129	51.29
13	w[1] = 0.597636, w[2] = 0.758869, w[3] = −0.307722, w[4] = 0.847564, w[5] = 0.542464, w[6] = 0.133098, w[7] = 0.930310, w[8] = 0.639165, w[9] = 0.726246, w[10] = 0.537192	0.5129	0	0.5129	0
14	w[1] = 0.071324, w[2] = 0.013643, w[3] = 0.395214, w[4] = −0.018087, w[5] = 0.091062, w[6] = 0.237512, w[7] = −0.047690, w[8] = 0.056467, w[9] = 0.025314, w[10] = 0.092948	0	0.5129	−0.5129	51.29
15	w[1] = 0.597636, w[2] = 0.758869, w[3] = −0.307722, w[4] = 0.847564, w[5] = 0.542464, w[6] = 0.133098, w[7] = 0.930310, w[8] = 0.639165, w[9] = 0.726246, w[10] = 0.537192	0.5129	0	0.5129	0
16	w[1] = 0.071324, w[2] = 0.013643, w[3] = 0.395214, w[4] = −0.018087, w[5] = 0.091062, w[6] = 0.237512, w[7] = −0.047690, w[8] = 0.056467, w[9] = 0.025314, w[10] = 0.092948	0	0.5129	−0.5129	51.29
17	w[1] = 0.597636, w[2] = 0.758869, w[3] = −0.307722, w[4] = 0.847564, w[5] = 0.542464, w[6] = 0.133098, w[7] = 0.930310, w[8] = 0.639165, w[9] = 0.726246, w[10] = 0.537192	0.5129	0	0.5129	0
18	w[1] = 0.071324, w[2] = 0.013643, w[3] = 0.395214, w[4] = −0.018087, w[5] = 0.091062, w[6] = 0.237512, w[7] = −0.047690, w[8] = 0.056467, w[9] = 0.025314, w[10] = 0.092948	0	0.5129	−0.5129	51.29
19	w[1] = 0.597636, w[2] = 0.758869, w[3] = −0.307722, w[4] = 0.847564, w[5] = 0.542464, w[6] = 0.133098, w[7] = 0.930310, w[8] = 0.639165, w[9] = 0.726246, w[10] = 0.537192	0.5129	0	0.5129	0
20	w[1] = 0.071324, w[2] = 0.013643, w[3] = 0.395214, w[4] = −0.018087, w[5] = 0.091062, w[6] = 0.237512, w[7] = −0.047690, w[8] = 0.056467, w[9] = 0.025314, w[10] = 0.092948	0	0.5129	−0.5129	51.29

all available performance assessment forms and fill the form with linguistic data needed to evaluate system architecture.

Access is gained to NFES using a valid user name and password. If the access right is granted, the home page is displayed. The home page is equipped with several hyperlinks that connect to the various system modules at the back-end. Clicking on the hyperlinks facilitate navigation between the various units of the system. The front-end interfaces are user-friendly.

4.2. NFES simulation

NFES is simulated using some sample life data obtained at Bells University of Technology (Bellstech) in Nigeria. The University has a DSS that is used to carry out all her operations. NFES being a web-based application was made available on-line for the users of the University's DSS to fill the DSSA performance assessment form (PAF) created for Bellstech. Twenty users were selected at random to fill and submit PAF on-line. The screen shot of the on-line version of PAF is presented in Fig. 13. It is also shown in Akinnuwesi et al. (2012, pp. 9337–9338). The screen shot of the simulation interface to input the parameters needed for simulation is presented in Fig. 14. The users filled PAF by rating each of the variables using any of the linguistic values (i.e. "*Strongly Agree*", "*Agree*", "*Not Sure*", "*Disagree*", "*Strongly Disagree*"). The quantitative linguistic weights assigned to the linguistic values are: *Strongly Agree = 5.00; Agree = 4.00; Not Sure = 3.00, Disagree = 2.00 and Strongly Disagree = 1.00*. Moreover each user indicates his/her rating confidence level for each variable rated. Rating confidence level ranges between 1 (Lowest level) – 10 (Highest level). At the inference engine, the linguistic values $\boldsymbol{x}_{i,i=1,2,3,\ldots n}$, go through the fuzzification process in the fuzzy engine and the crisp values generated are passed to the matching engine where the quantitative value for each software construct, $\boldsymbol{y}_{j,j=1,2,3\ldots m}$, is computed. The computed values of $\boldsymbol{y}_{j,j=1,2,3\ldots m}$, are passed to the neural network engine where the simulation process of computing the DSSA performance value takes place. The neural network process converges at a performance value of ***0.5129 (51.29%)*** using a *Learning Rate = 0.70* and *Initial Synaptic Weight = 0.21*. The *Learning Rate* and *Initial Synaptic Weight values* were varied with the view of enabling the neural network engine to oscillate around a constant value that will be taken as the performance value of the DSSA in question. Some intermediate results are presented in Appendix A while the simulation results from the neural network engine is presented in Table 3.

5. Conclusion

This research focused on building an expert system that addresses the limitations identified in previous DSSA performance evaluation models. Some of the drawbacks are: (1) none usage of user-centric parameters to measure performance of distributed software architectures but rather use machine-centric variables for measurement; (2) non-utilization of subjective values for DSSA performance measurement but rather objective values drawn from the machine. Though Akinnuwesi (2011) and Akinnuwesi et al. (2012) designed a user-centric neuro-fuzzy algorithm (NFPEM) to address the aforementioned drawbacks, the NFPEM but it is not flexible and scalable enough. Also it was not designed as an expert system. NFES is an improvement on NFPEM in Akinnuwesi (2011) and Akinnuwesi et al. (2012) and ESSE in Vlahavas et al. (1999). The developed expert system removes the rigidity observed in our previous works. It could also evaluate performance of systems at architectural level as against ESSE, which evaluates performance at system implementation level. Therefore, with NFES, a performance engineer is now able to define organizational variables and matching function peculiar to any given organization whose DSSA is to be measured for performance. This is done by interacting with the management of the client organization and the end users in order to get all the details about organizational issues and factors that motivate and support the development of DSSA that the client organization intends to implement. Moreover the use of threshold performance value is optional, thus the neural network engine is now trained with different varying values for ***Learning Rate (LR)*** and ***Initial Synaptic Weight (ISW)***. At a given value of LR and ISW, after some iteration, the outputs of the neural network engine oscillate about two points with constant values (i.e. $\mathbf{P_n} = 0.0$ (lower point) and $\mathbf{P_n} > 0.0$ (upper point)). At this point the neural network engine has converged and the value of $\mathbf{P_n}$ at the upper point is taken as the performance value of the DSSA.

It is worth noting that a good DSSA is a function of the collaborative efforts of the client organization and software engineer. Thus the success/failure of an enterprise application system depends on the cordiality between the end users and the software developers (Brian, Jerry, & Edward, 2006; Muthitacharoen & Saeed, 2009; Procaccino & Verner, 2009; Serkan & Kursat, 2005). Therefore evaluating software system at architectural level using NFES will help to produce a performance value that indicates the extent to which the DSSA will respond to the users' requirements if it is implemented. In the long run, this guides the management of the client organization to fund only software system whose architecture has been determined to respond well to the organizational requirements and operations.

Presented in Table 4 is the comparison of NFES with ESSE (Vlahavas et al., 1999), GPSE (Behrouz et al., 2009) and NFPEM (Akinnuwesi, 2011; Akinnuwesi et al., 2012).

Table 4
Comparison of NFES with ESSE, GPSE and NFPEM models.

	ESSE (Vlahavas et al., 1998)	GPSE (Behrouz et al., 2009)	NFPEM (Akinnuwesi, 2011; Akinnuwesi et al., 2012)	NFES
Web-based performance evaluation system				✓
Use softcomputing approach			✓	✓
Uses software architectural components			✓	✓
Use organizational contextual variables	✓	✓	✓	✓
Support dynamic definition of architectural components and organizational variables				✓
Support infinite number of organizational variables				✓
Supports infinite number of architecture components				✓
Supports dynamic definition of component–variables relationship (i.e. Matching function)				✓
Automatic computation of performance				✓
Uses organizational variables as performance metrics			✓	✓
Evaluate software performance at architectural level			✓	✓

Appendix A. Intermediate computations (sample)

Legend	
LMV	Lower bound membership value
MMV	Median point membership value
UMV	Upper bound membership value
LRV	Lower bound rated value
MRV	Median point rated value
URV	Upper bound rated value
RC	Rating confidence
NRC	Normalized rating confidence

The DSS of your organization satisfies all communication rules that are established to relate with external organizations

User	Value	RC	NRC	Crisp	LMV	MMV	UMV	LRV	MRV	URV
Crisp value (Mean): 3.05317693801										
1	5	1	1	4.769	0.860	1.000	0.000	4.500	5.000	5.500
2	5	1	1	4.769	0.860	1.000	0.000	4.500	5.000	5.500
3	4	1	1	4.047	0.625	1.000	0.860	3.500	4.000	4.500
4	5	1	1	4.769	0.860	1.000	0.000	4.500	5.000	5.500
5	5	1	1	4.769	0.860	1.000	0.000	4.500	5.000	5.500
6	4	1	1	4.047	0.625	1.000	0.860	3.500	4.000	4.500
7	4	0.9	0.7	2.768	0.710	0.640	0.538	2.450	2.800	3.150
8	4	0.9	0.7	2.768	0.710	0.640	0.538	2.450	2.800	3.150
9	4	0.9	0.7	2.768	0.710	0.640	0.538	2.450	2.800	3.150
10	4	0.9	0.7	2.768	0.710	0.640	0.538	2.450	2.800	3.150
11	5	0.9	0.7	3.533	0.538	0.625	0.713	3.150	3.500	3.850
12	4	0.9	0.7	2.768	0.710	0.640	0.538	2.450	2.800	3.150
13	4	0.9	0.7	2.768	0.710	0.640	0.538	2.450	2.800	3.150
14	4	0.9	0.7	2.768	0.710	0.640	0.538	2.450	2.800	3.150
15	4	0.9	0.7	2.768	0.710	0.640	0.538	2.450	2.800	3.150
16	5	0.9	0.7	3.533	0.538	0.625	0.713	3.150	3.500	3.850
17	4	0.9	0.7	2.768	0.710	0.640	0.538	2.450	2.800	3.150
18	5	0.8	0.15	0.735	0.325	0.250	0.175	0.675	0.750	0.825
19	4	0.8	0.15	0.591	0.475	0.400	0.325	0.525	0.600	0.675
20	4	0.8	0.15	0.591	0.475	0.400	0.325	0.525	0.600	0.675
The DSS of your organization satisfies the laid down communications rules and semantics for the units within the organization to relate										
Crisp value (Mean): 2.96437318719										
1	4	1	1	4.047	0.625	1.000	0.860	3.500	4.000	4.500
2	5	1	1	4.769	0.860	1.000	0.000	4.500	5.000	5.500
3	4	1	1	4.047	0.625	1.000	0.860	3.500	4.000	4.500
4	4	1	1	4.047	0.625	1.000	0.860	3.500	4.000	4.500
5	5	1	1	4.769	0.860	1.000	0.000	4.500	5.000	5.500
6	4	1	1	4.047	0.625	1.000	0.860	3.500	4.000	4.500
7	4	1	1	4.047	0.625	1.000	0.860	3.500	4.000	4.500
8	4	1	1	4.047	0.625	1.000	0.860	3.500	4.000	4.500
9	4	0.9	0.6	2.383	0.780	0.720	0.660	2.100	2.400	2.700
10	4	0.9	0.6	2.383	0.780	0.720	0.660	2.100	2.400	2.700
11	4	0.9	0.6	2.383	0.780	0.720	0.660	2.100	2.400	2.700
12	4	0.9	0.6	2.383	0.780	0.720	0.660	2.100	2.400	2.700
13	5	0.9	0.6	2.989	0.660	1.000	0.575	2.700	3.000	3.300
14	5	0.9	0.6	2.989	0.660	1.000	0.575	2.700	3.000	3.300
15	5	0.9	0.6	2.989	0.660	1.000	0.575	2.700	3.000	3.300
16	5	0.9	0.6	2.989	0.660	1.000	0.575	2.700	3.000	3.300
17	5	0.8	0.2	1.071	0.100	0.000	0.580	0.900	1.000	1.100
18	5	0.8	0.2	1.071	0.100	0.000	0.580	0.900	1.000	1.100
19	4	0.8	0.2	0.767	0.300	0.200	0.100	0.700	0.800	0.900
20	5	0.8	0.2	1.071	0.100	0.000	0.580	0.900	1.000	1.100

(*continued on next page*)

The DSS of your organization provides friendly features that gears the willingness of the users to embrace its usage
Crisp value (Mean): 3.20610264493

1	5	1	1	4.769	0.860	1.000	0.000	4.500	5.000	5.500
2	5	1	1	4.769	0.860	1.000	0.000	4.500	5.000	5.500
3	5	1	1	4.769	0.860	1.000	0.000	4.500	5.000	5.500
4	4	1	1	4.047	0.625	1.000	0.860	3.500	4.000	4.500
5	5	1	1	4.769	0.860	1.000	0.000	4.500	5.000	5.500
6	5	1	1	4.769	0.860	1.000	0.000	4.500	5.000	5.500
7	4	1	1	4.047	0.625	1.000	0.860	3.500	4.000	4.500
8	4	1	1	4.047	0.625	1.000	0.860	3.500	4.000	4.500
9	5	1	1	4.769	0.860	1.000	0.000	4.500	5.000	5.500
10	4	0.9	0.55	2.242	0.415	0.760	0.705	1.925	2.200	2.475
11	5	0.9	0.55	2.721	0.705	0.650	0.506	2.475	2.750	3.025
12	5	0.9	0.55	2.721	0.705	0.650	0.506	2.475	2.750	3.025
13	4	0.9	0.55	2.242	0.415	0.760	0.705	1.925	2.200	2.475
14	5	0.9	0.55	2.721	0.705	0.650	0.506	2.475	2.750	3.025
15	4	0.9	0.55	2.242	0.415	0.760	0.705	1.925	2.200	2.475
16	5	0.9	0.55	2.721	0.705	0.650	0.506	2.475	2.750	3.025
17	4	0.9	0.55	2.242	0.415	0.760	0.705	1.925	2.200	2.475
18	5	0.9	0.55	2.721	0.705	0.650	0.506	2.475	2.750	3.025
19	4	0.8	0.1	0.397	0.650	0.600	0.550	0.350	0.400	0.450
20	4	0.8	0.1	0.397	0.650	0.600	0.550	0.350	0.400	0.450

The DSS of your organization supports the IT infrastructure that are available in the organization
Crisp value (Mean): 2.9763001086

1	4	1	1	4.047	0.625	1.000	0.860	3.500	4.000	4.500
2	4	1	1	4.047	0.625	1.000	0.860	3.500	4.000	4.500
3	4	1	1	4.047	0.625	1.000	0.860	3.500	4.000	4.500
4	5	1	1	4.769	0.860	1.000	0.000	4.500	5.000	5.500
5	5	1	1	4.769	0.860	1.000	0.000	4.500	5.000	5.500
6	5	1	1	4.769	0.860	1.000	0.000	4.500	5.000	5.500
7	5	1	1	4.769	0.860	1.000	0.000	4.500	5.000	5.500
8	4	0.9	0.65	2.579	0.745	0.680	0.615	2.275	2.600	2.925
9	4	0.9	0.65	2.579	0.745	0.680	0.615	2.275	2.600	2.925
10	5	0.9	0.65	3.255	0.615	0.563	0.644	2.925	3.250	3.575
11	4	0.9	0.65	2.579	0.745	0.680	0.615	2.275	2.600	2.925
12	5	0.9	0.65	3.255	0.615	0.563	0.644	2.925	3.250	3.575
13	4	0.9	0.65	2.579	0.745	0.680	0.615	2.275	2.600	2.925
14	5	0.9	0.65	3.255	0.615	0.563	0.644	2.925	3.250	3.575
15	5	0.9	0.65	3.255	0.615	0.563	0.644	2.925	3.250	3.575
16	5	0.8	0.25	1.246	0.575	0.550	0.525	1.125	1.250	1.375
17	5	0.8	0.25	1.246	0.575	0.550	0.525	1.125	1.250	1.375
18	4	0.8	0.25	1.033	0.125	1.000	0.575	0.875	1.000	1.125
19	5	0.8	0.25	1.246	0.575	0.550	0.525	1.125	1.250	1.375
20	4	0.7	0.05	0.199	0.825	0.800	0.775	0.175	0.200	0.225

The DSS of your organization is developed within the limit of the organization's budget for it
Crisp value (Mean): 3.11224098449

1	4	1	1	4.047	0.625	1.000	0.860	3.500	4.000	4.500
2	4	1	1	4.047	0.625	1.000	0.860	3.500	4.000	4.500
3	5	1	1	4.769	0.860	1.000	0.000	4.500	5.000	5.500
4	4	1	1	4.047	0.625	1.000	0.860	3.500	4.000	4.500
5	5	1	1	4.769	0.860	1.000	0.000	4.500	5.000	5.500
6	5	1	1	4.769	0.860	1.000	0.000	4.500	5.000	5.500
7	5	0.9	0.7	3.533	0.538	0.625	0.713	3.150	3.500	3.850
8	5	0.9	0.7	3.533	0.538	0.625	0.713	3.150	3.500	3.850
9	4	0.9	0.7	2.768	0.710	0.640	0.538	2.450	2.800	3.150
10	4	0.9	0.7	2.768	0.710	0.640	0.538	2.450	2.800	3.150
11	4	0.9	0.7	2.768	0.710	0.640	0.538	2.450	2.800	3.150
12	4	0.9	0.7	2.768	0.710	0.640	0.538	2.450	2.800	3.150
13	4	0.9	0.7	2.768	0.710	0.640	0.538	2.450	2.800	3.150
14	4	0.9	0.7	2.768	0.710	0.640	0.538	2.450	2.800	3.150
15	5	0.9	0.7	3.533	0.538	0.625	0.713	3.150	3.500	3.850

16	5	0.9	0.7	3.533	0.538	0.625	0.713	3.150	3.500	3.850
17	5	0.9	0.7	3.533	0.538	0.625	0.713	3.150	3.500	3.850
18	4	0.8	0.15	0.591	0.475	0.400	0.325	0.525	0.600	0.675
19	5	0.8	0.15	0.735	0.325	0.250	0.175	0.675	0.750	0.825
20	4	0.7	0.05	0.199	0.825	0.800	0.775	0.175	0.200	0.225

The feasibility study done by the DSS project team in your organization is adequate
Crisp value (Mean): 3.14283058493

1	4	1	1	4.047	0.625	1.000	0.860	3.500	4.000	4.500
2	5	1	1	4.769	0.860	1.000	0.000	4.500	5.000	5.500
3	4	1	1	4.047	0.625	1.000	0.860	3.500	4.000	4.500
4	5	1	1	4.769	0.860	1.000	0.000	4.500	5.000	5.500
5	4	1	1	4.047	0.625	1.000	0.860	3.500	4.000	4.500
6	5	0.9	0.75	3.785	0.594	0.688	0.785	3.375	3.750	4.125
7	5	0.9	0.75	3.785	0.594	0.688	0.785	3.375	3.750	4.125
8	4	0.9	0.75	2.987	0.675	1.000	0.594	2.625	3.000	3.375
9	5	0.9	0.75	3.785	0.594	0.688	0.785	3.375	3.750	4.125
10	4	0.9	0.75	2.987	0.675	1.000	0.594	2.625	3.000	3.375
11	4	0.9	0.75	2.987	0.675	1.000	0.594	2.625	3.000	3.375
12	5	0.9	0.75	3.785	0.594	0.688	0.785	3.375	3.750	4.125
13	5	0.9	0.75	3.785	0.594	0.688	0.785	3.375	3.750	4.125
14	5	0.9	0.75	3.785	0.594	0.688	0.785	3.375	3.750	4.125
15	5	0.9	0.75	3.785	0.594	0.688	0.785	3.375	3.750	4.125
16	4	0.9	0.75	2.987	0.675	1.000	0.594	2.625	3.000	3.375
17	5	0.8	0.2	1.071	0.100	0.000	0.580	0.900	1.000	1.100
18	5	0.8	0.2	1.071	0.100	0.000	0.580	0.900	1.000	1.100
19	4	0.7	0.1	0.397	0.650	0.600	0.550	0.350	0.400	0.450
20	4	0.3	0.05	0.199	0.825	0.800	0.775	0.175	0.200	0.225

References

Akinnuwesi B. A (2011*). Development of a user-centric model for evaluating distributed software system architecture using neuro-fuzzy technique.* Ph.D. Thesis submitted to the Postgraduate school, Ladoke Akintola University of Technology, Ogbomosho, Nigeria.

Akinnuwesi, B. A., Uzoka, F. M. E., Olabiyisi, S. O., & Omidiora, E. O. (2012). A framework for user-centric model for evaluating the performance of distributed software system architecture. *International Journal of Expert Systems with Applications, 39*(10), 9323–9339 [Elsevier].

Behrouz, H. F., Vani, M., & Abdel-Halim, E. (2009). A general purpose software evaluation system. *Informatica, 33*, 261–270.

Brian, W., Jerry, F., & Edward, M. (2006). The web of system performance. *Communications of the ACM, 49*(5), 93–99.

Cannolly, T., & Begg, C. (2002). *Database system: a practical approach to design, implementation and management* (3rd ed.). Edinburgh Gate, Harlow, Essex CM20 2JE: Publication of Pearson Education Limited, Addison Wesley.

Kroenke, M. D. (1992). *Database processing – Fundamentals, design and implementation (pp. 1–430)* (4th ed.). New York: Macmillan Publishing Company.

Lloyd, G. W., & Connie, U. S. (1998). Performance evaluation of distributed software architectures. In *Proceedings of international CMG conference* (pp. 337–345).

Muthitacharoen, A., & Saeed, A. K. (2009). Examining user involvement in continuous software development (a case of error reporting system). *Communications of the ACM, 52*(9), 113–117.

Olabiyisi, S. O., Omidiora, E. O., Uzoka, F. M. E., Victor, M., & Akinnuwesi, B. A, (2010), A survey of performance evaluation models for distributed software system architecture. In *Proceedings of the World congress on engineering and computer science WCECS 2010, October 20–22* (Vol. I): San Francisco, USA.

Olabiyisi, S. O., Omidiora, E. O., Uzoka, F. M. E., Akinnuwesi, B. A., Mbarika, V. W., Kourouma, M. K., et al. (2011). Exploratory study of performance evaluation models for distributed software architecture.. *Journal of Computer Resource Management (International Journal of Computer Measurement Group Inc), Autumn* (130), 47–57.

Procaccino, J. D., & Verner, M. J. (2009). Software developers' views of end-users and project success. *Communications of the ACM, 52*(5), 113–116.

Samuel, K. (2006). Performance modeling and evaluation of distributed component-based system using queuing petri-nets. *IEEE Transactions of Software Engineering, 32*(7), 487–502.

Samuel, K., & Alejandro, B. (2003). performance modeling of distributed e-business applications using queuing petri-nets. In *Proceedings of IEEE international symposium on performance analysis of systems and software, March* (pp. 145–153).

Serkan, C. A. K., & Kursat, C. (2005). Bridging the gap between users and developers in software intensive projects of turkish defense industry. Available @ <http://www.metu.edu.tr/_kursat/Cak&Cagiltay.pdf>. Accessed date: October 26, 2012.

Simonetta, B., Roberto, M., & Moreno, M. (2004). Performance evaluation of software architecture with queuing networking models. In *Proceedings of ESMC'04 October 25–27*: Paris France.

Vlahavas, I., Stamelos, I., Refanidis, I., & Tsoukias, A. (1999). ESSE: an expert system for software evaluation. *Knowledge Based Systems, Elsevier, 4*(12), 183–197.