Evaluation of Software Architectures using Multicriteria Fuzzy Decision Making Technique

G. Zayaraz
Dept of Computer Sci. Engg.
Pondicherry Engg. College
Puducherry– 605 014, India
gzayaraz@pec.edu

S. Vijayalakshmi
Dept. of Info. Tech.
Christ College of Engg. & Technology
Puducherry– 605 010, India
sviji7195@yahoo.com

V. Vijayalakshmi
Dept. of Elec & Comm. Engg.
Pondicherry Engg. College
Puducherry– 605 014, India
vvijizai@pec.edu

Abstract- **Software architectures is a critical aspect in the design and development of software. Architecture of software is a collection of design decisions that are expensive to change. A correct architecture has the largest single impact on cost and quality of the product. Though architecting constitutes 10% of the product development cycle, it determines 90% of the product development costs. Given the impact that software architecture has on a project's success, the need to choose the right architecture assumes significance. Organizations often need to choose software architecture for future development from several competing candidate architectures. In this paper, a new architecture selection method based on multicriteria fuzzy decision making technique has been developed and validated using a suitable case study.**

Keywords - Quality attributes; Software architecture; Decision making; Fuzzy Decision Making.

I. INTRODUCTION

Software architectures are abstract design artifacts of the software system to be developed. They are usually constructed from the functional and nonfunctional requirements of the software system. Choice of which alternative architecture to go with is a crucial part in any software development as this choice affects the quality of final software product. Conventional qualitative architecture evaluation techniques discussed and quantitative selection techniques in [1] have been analyzed to identify their limitations. In order to overcome the limitations and challenges, a quantitative evaluation method based on multicriteria fuzzy decision making is proposed. The existing evaluation method provides the rationale for architecture selection process by measuring the conformance to requirements of each candidate architecture.

Architecture reflects the functional and nonfunctional requirements of a software system. Architectures are the only artifact available at the early stages of the software development. Architectures both newly developed and reused have to be rigorously evaluated for its conformance to requirements, as it directly affects the quality of final software product. Choice on architecture alternatives is made on the basis of stakeholders' expectations and preferences. Stakeholders are responsible for making crucial design decisions. Requirements of stakeholders are discrete and preferences may also vary. The selected architecture may undergo small changes (architectural degeneration) in later phases of the software life cycle which leads to repetition of the entire evaluation. Thus, the process of architecture evaluation is a complex task.

Quantitative evaluation technique used for selection of architecture has to be systematic and based on statistical methods. It also has to provide a mechanism for verifying the structural changes in architectural for conformance with requirements. In this research work, an attempt has been made to propose a quantitative evaluation method based on multicriteria fuzzy decision making technique which selects an architectural based on the requirements of stakeholders.

II.DESCRIPTION OF THE MULTICRITERIA FUZZY DECISION MAKING

In multicriteria decision problems, the architectures are evaluated according to a number of quality attributes. Each quality attribute induces a particular ordering of the architecture and what is needed is a procedure to construct an overall preference ranking. The basic information involved in multicriteria decision making problem can be expressed by the matrix [2].

$$R = \begin{array}{c} \\ X1 \\ X2 \\ . \\ Xm \end{array} \begin{array}{c} \begin{array}{cccc} C1 & C2 & \dots & Cn \end{array} \\ \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ . & & & \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix} \end{array} \quad (1)$$

where X = (X1, X2,…, Xm) is finite set of architecture and C=(C1, C2,…, Cn) is the set of quality attributes. Each entry in the matrix are assumed to be real numbers [0 1], expressing the degree to which architectures Xi satisfies the quality attributes Cj. Thus R can be viewed as a matrix representation of a fuzzy relation on X × C.

If it happens that, instead of R, an architecture matrix R'= [aij'], whose entries are arbitrary real numbers, then R'= [aij'] can be converted to the desired matrix R by the formula.

$$a_{ij} = \frac{a_{ij}' - \min_{i \in N_m} a_{ij}'}{\max_{i \in N_m} a_{ij}' - \min_{i \in N_m} a_{ij}'} \quad (2)$$

for all $i \in N_m$ and $j \in N_m$.

The common approach in multicriteria decision problem is to convert them into single criterion decision problems. For this need to find a global quality attribute

$r_i = h(a_{i1}, a_{i2}, \ldots a_{in})$

that for each $X_i \in X$ is an adequate aggregate of values ai1, ai2,…, ain to which the individual quality attributes C1, C2,…, Cn are satisfied.

The most commonly employed aggregating operator is the preference coefficient average

$$r_j = \frac{\sum_{j=1}^{n} w_j a_{ij}}{\sum_{j=1}^{n} w_j} \quad (3)$$

where w1, w2,…,wn are preference coefficients that indicate the relative importance of quality attributes C1, C2,…,Cn.

When the quality attributes are split in to two groups 'benefit' and 'cost', the elements of the matrix R' can be normalized [3] using the formulae

$r_{ij} = \frac{a_{ij}}{\max_k [a_{kj}]}$ for the benefit quality attributes C_j

$r_{ij} = \frac{\min_k [a_{kj}]}{a_{ij}}$ for the cost quality attributes C_j

and thus we get the normalized matrix R=(rij)m x n.

Based on the normalized decision matrix R, the overall aggregate preference value of the architecture Xi can be expressed by the additive preference coefficient averaging operator as

$$z_i(w) = \sum_{j=1}^{n} r_{ij} w_j \quad ; i = 1,2,\ldots,m \quad (4)$$

The greater the overall attribute value zi (w), the better is the corresponding architecture Xi will be. Even though the quality attributes are split into two groups unlike the MCDA methods, the total preference of all the quality attributes should add up to 1 instead preference coefficients of benefit quality attributes and cost quality attributes being separately adding to 1.

III.CASE STUDY

A case study of real-time stock monitoring system [4] is taken and evaluated using the proposed quantitative evaluation method. The main objective of case study is to validate the proposed evaluation method. Input to the evaluation method is set of candidate architectures and its quality characteristics measured. Expected outputs are total satisfaction value for each candidate architecture. Architectures with highest total satisfaction value is selected for further phases of the software development life cycle.

The primary goal of real-time stock monitoring system is to capture, analyze and broadcast stock events information in real-time. It is a soft real-time system where some of the events may miss their deadline without affecting the whole system behavior. The system is a real-time data provider for monitoring stocks of small and medium size stock exchanges for brokers and independent investors. An antenna (feed server), external to the system, provides the data (feed) to the data server. A feed contains the relevant information of a stock exchange transaction. Feeds are supposed to be reliable and available.

The clients, namely the brokers are distributed in different geographical locations and subscribed to the data server. When a change on the feed to which a client has subscribed occurs, the feed is broadcasted to him/her by the data server, according to a strict time delay. The time delay will depend on the network structure used to send the information to the clients. The type of service offered depends on this delay. Internet facilities through commercial browsers are required for the system. The publisher/subscriber stores the client subscriptions, the actual values in the client subscription DB and the data server respectively. Three different architectural solutions are available for real-time stock monitoring system namely publisher/subscriber pattern, repository pattern and broadcast pattern.

A. *Inputs for Evaluation*

Inputs to the evaluation method are set of candidate architectures along with their quality characteristics measured. There are three candidate architectures considered namely publisher/subscriber pattern (A), repository pattern (B) and broadcast pattern (C).

Publisher/Subscriber Pattern (A)

In this type of candidate architecture, clients register their interest for stocks with the subscriber. The subscriber records the details of the clients in the database. A change in stock prices causes the publisher to notify these changes to the interested clients. Publisher/Subscriber pattern is shown in Figure 1.

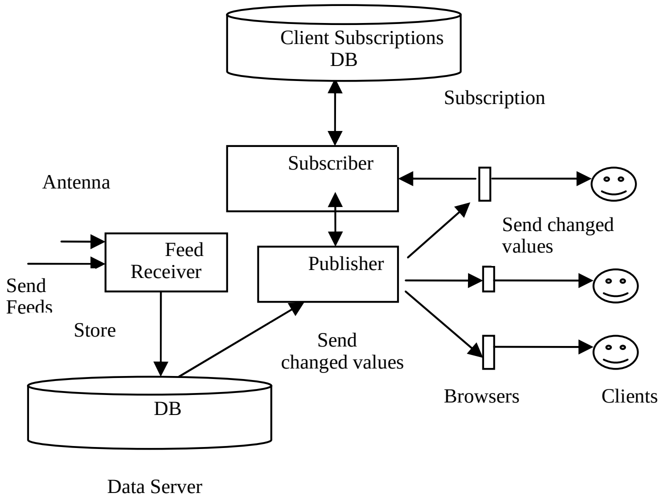

Figure 1. Publisher/Subscriber Pattern (A)

Repository Pattern (B)

In this type of candidate architecture, clients request the server for data about the stocks. Requests by clients may or may not be done periodically. Usage of proper queuing mechanism helps to avoid conflicts among requesting clients. A repository pattern is shown in Figure 2.

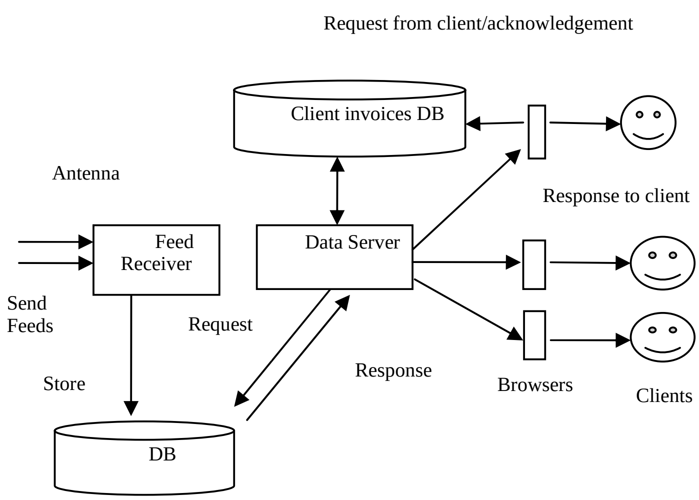

Figure 2: Repository Pattern (B)

Broadcast Pattern (C)

In this type of candidate architecture, a change in stock prices causes the server to broadcast these changes to their clients. Communication between clients and server is unidirectional. Broadcast pattern is shown in Figure 3.

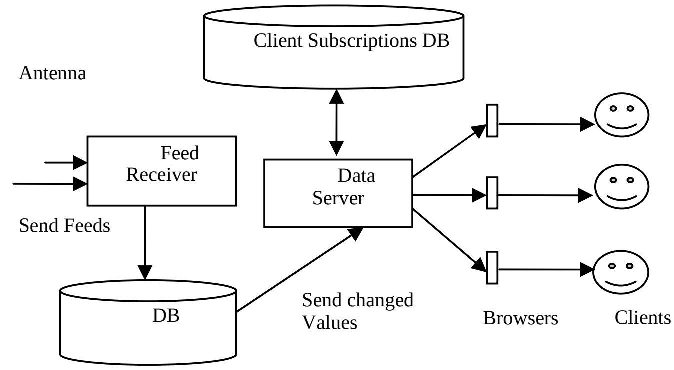

Figure 3: Broadcast Pattern(C)

B. Quality Attributes

The real-time stock monitoring system is identified with eight quality attributes [5] namely response time, learnability, maintainability, recoverability, reusability, cost, development time and team size. Architecture evaluation is carried out for these quality attributes for conformance with stakeholders' requirements. Measured values of candidate architectures are listed in Table I.

Response Time

Response time is defined as the time required for completing a transaction. It is the sum of processing time, queuing time and data transfer time. It is measured in milliseconds (ms). In the case of repository pattern, the response time is high. This is due to the usage of queuing mechanism to handle simultaneous request from the clients. Moreover, each time to access information about stocks from the server, clients make a request to the server and in turn they receive acknowledgement. The requested data is sent by the server to the clients and it turns the server received acknowledgement from the clients. However, in the case of subscriber pattern data is sent to all clients periodically provided the interested clients register themselves once with the server.

Learnability

Learnability is defined as the time required by the user to understand the software and work with it. This includes the training period given for the users. It is measured in hours (hrs).

Maintainability

Maintainability is defined as the time taken to make successful modifications in the architecture. It is measured in hours (hrs). Maintainability mainly depends on the number of components and their interactions required for achieving the functionalities of the system. The subscriber/publisher has the highest number of components.

Recoverability

Recoverability is defined as the time taken to recover from failure state to working state. It is measured in seconds (secs). In repository structure, when the client side fails and recovers it can quickly reestablish the current status by requesting the server. However, in other two structures clients have to wait for the next periodic information.

Reusability

Reusability is defined as the number of components and connectors that can be reused. It is measured in number (nos).

Cost

It is the cost associated with developing the software product. It is measured in rupees (rs). The cost for repository system is low because it is built by using the existing components.

Development Time

This is the measure of time taken to build the software system. Its unit of measurement is weeks (wks). Since the repository pattern reuses existing components, its development time is less.

Team Size

It is defined as the number of technical persons required to build the project. Its unit of measurement in numbers (nos).

TABLE I. MEASURED VALUES

Architecture	**Response time (ms)**	**Maintainability (sec)**	**Learnability (hrs)**	**Reusability (nos)**	**Recoverability (secs)**	**Cost (Rs in Lacs)**	**Team size (nos)**	**Development Time (wks)**
A	10	200	5	1	20	8	20	60
B	20	25	8	5	10	4	10	30
C	12	200	3	1	5	6	5	20

C. *Multicriteria Fuzzy Decision Making*

The multicriteria fuzzy decision making technique is applied to this problem and analyzed under multicriteria fuzzy decision making technique.

Data

W_1	W_2	W_3	W_4	W_5	W_6	W_7	W_8
0.1	0.05	0.2	0.15	0.1	0.2	0.15	0.05

$$\begin{matrix} A \\ B \\ C \end{matrix} \begin{pmatrix} 10 & 5 & 1 & 200 & \vdots & 20 & 8 & 20 & 60 \\ 20 & 8 & 5 & 25 & \vdots & 10 & 4 & 10 & 30 \\ 12 & 3 & 1 & 200 & \vdots & 5 & 6 & 5 & 20 \end{pmatrix}$$

The normalized matrix is

W_1	W_2	W_3	W_4	W_5	W_6	W_7	W_8
0.1	0.05	0.2	0.15	0.1	0.2	0.15	0.05

$$\begin{pmatrix} \frac{1}{2} & \frac{5}{8} & \frac{1}{5} & 1 & \vdots & \frac{1}{4} & \frac{1}{2} & \frac{1}{4} & \frac{1}{3} \\ 1 & 1 & 1 & \frac{25}{200} & \vdots & \frac{1}{2} & 1 & \frac{1}{2} & \frac{1}{2} \\ \frac{6}{10} & \frac{3}{8} & \frac{1}{5} & 1 & \vdots & 1 & \frac{2}{3} & 1 & 1 \end{pmatrix}$$

$$Z_1 = \frac{1}{2}\times 0.1+\frac{5}{8}\times 0.05+\frac{1}{5}\times 0.2+1\times 0.1$$

$$+\frac{1}{4}\times 0.1+\frac{1}{2}\times 0.2+\frac{1}{4}\times 0.15+\frac{1}{3}\times 0.05$$

$$= 0.3975$$

$$Z_2 = 1\times 0.1+1\times 0.05+1\times 0.2+\frac{25}{200}\times 0.1$$

$$+\frac{1}{2}\times 0.1+1\times 0.2+\frac{1}{2}\times 0.15+\frac{1}{2}\times 0.05$$

$$= 0.712$$

$$Z_3 = \frac{6}{10}\times 0.1+\frac{3}{8}\times 0.05+\frac{1}{5}\times 0.2+1\times 0.1$$

$$+1\times 0.1+\frac{2}{3}\times 0.2+1\times 0.15+1\times 0.05$$

$$= 0.6505$$

From the values of Zi, the ranking of the architectures is B > C > A.

It is to be observed that the proposed multi criteria fuzzy decision making method gives the result consistent. but the computation involved is simpler than the other methods.

IV.CONCLUSION

Thus a simple and efficient architecture selection method based on multicriteria fuzzy decision making technique has been developed and validated using the stock monitoring system case study. The proposed method can handle the uncertainty and vagueness in stakeholders' requirements.

REFERENCES

[1] G. Zayaraz and Dr. P. Thambidurai, "Quantitative Model for the Evaluation of Software Architectures", Journal of Software Quality Professional, American Society for Quality, Vol.9, no.3, pp. 28-40, June 2007.

[2] J. George, Klir and Bo Yuan, Fuzzy Sets and Fuzzy Logic Theory and Applications, Prentice-Hall, pp. 399-401, 1997.

[3] Zeshui Xu, "Multiple-Attribute Group Decision Making with Different Formats of Preference Information on Attributes", IEEE Transaction on Systems, Man, and Cybernetics-Part B:Cybernetics, Vol.37, No.6, December 2007.

[4] F. Losavio, L. Chirinos, N. Levy and A. Ramdane, "Quality Characteristics of Software Architecture", Journal of Object Technology, Vol. 2, no.2, pp. 133-150, March 2003.

[5] M. Stahlberg, C. Wohlin, L. Lundberg and M. Mattsson, "A Method for understanding Quality Attributes in Software Architecture Structures", Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering, pp. 819-826, July 2002.