01 Oct 2010

# Technical Target Setting in QFD for Web Service Systems using an Artificial Neural Network

Lianzhang Zhu

Xiaoqing Frank Liu
*Missouri University of Science and Technology*, fliu@mst.edu



## Recommended Citation

L. Zhu and X. F. Liu, "Technical Target Setting in QFD for Web Service Systems using an Artificial Neural Network," *IEEE Transactions on Services Computing*, Institute of Electrical and Electronics Engineers (IEEE), Oct 2010.
The definitive version is available at https://doi.org/10.1109/TSC.2010.45

# Technical Target Setting in QFD for Web Service Systems Using an Artificial Neural Network

Lianzhang Zhu and Xiaoqing (Frank) Liu, *Member*, *IEEE*

**Abstract**—There are at least two challenges with quality management of service-oriented architecture based web service systems: 1) how to link its technical capabilities with customer's needs explicitly to satisfy customers' functional and nonfunctional requirements; and 2) how to determine targets of web service design attributes. Currently, the first issue is not addressed and the second one is dealt with subjectively. Quality Function Deployment (QFD), a quality management system, has found its success in improving quality of complex products although it has not been used for developing web service systems. In this paper, we analyze requirements for web services and their design attributes, and apply the QFD for developing web service systems by linking quality of service requirements to web service design attributes. A new method for technical target setting in QFD, based on an artificial neural network, is also presented. Compared with the conventional methods for technical target setting in QFD, such as benchmarking and the linear regression method, which fail to incorporate nonlinear relationships between design attributes and quality of service requirements, it sets up technical targets consistent with relationships between quality of web service requirements and design attributes, no matter whether they are linear or nonlinear.

**Index Terms**—Web service system, service quality management, Bayesian regularized neural network, quality function deployment (QFD), technical targets setting.

## 1 Introduction

Service-oriented Architecture (SOA), representing a technology and interaction framework in which systems are modular and loosely coupled, is becoming popular in the industry, and is increasingly accepted as a mainstream technology in the IT world [1], [2]. It provides an evolutionary way for companies to organize their information flow effectively and to share resources with their stakeholders efficiently through a highly flexible infrastructure. It can reduce the development time and enables interoperability. An enterprise software customer survey from McKinsey & Company and Sand Hill Group examined the strategies of 475 senior IT and business executives in 2007 [3], and its result showed that SOA is a top-of-mind issue for software customers. Another survey in 2008 [4], which included more than 850 enterprise software customers, showed that more than 60 percent of respondents thought the innovation was likely driven by major trends, such as SOA. It is becoming a trend for system development and integration where systems group functionality around business processes. Many IT giants such as Google, Amazon, Microsoft, IBM and salesforce.com followed this trend, and provided their SOA-based web platforms to the software industry [5], [6].

- *L. Zhu is with the College of Computer and Communication Engineering, China University of Petroleum (East China), #739 Beiyi Road, Dongying City 257061, Shandong, P.R. China. E-mail: zhulz@upc.edu.cn.*
- *X.F. Liu is with the Computer Science Department, Missouri University of Science and Technology, 325 Computer Science Bldg., 500 W. 15th St., Rolla, MO 65409. E-mail: fliu@mst.edu.*

*Manuscript received 11 Sept. 2009; revised 9 Jan. 2010; accepted 27 May 2010; published online 1 Sept. 2010.*
*For information on obtaining reprints of this article, please send e-mail to: tsc@computer.org, and reference IEEECS Log Number TSC-2009-09-0192.*
*Digital Object Identifier no. 10.1109/TSC.2010.45.*

In the development of SOA-based web service system platforms, the customer's requirements drive their designs. Therefore, we should explicitly elicit and specify customers' requirements and trace them do design attributes [7].

Quality Function Deployment (QFD), one of the major quality systems, is a methodology for incorporating the voice of customers, both spoken and unspoken, into a product development process. It has been applied successfully to develop numerous products such as automobiles, aircraft, and consumer electronics, and to improve their quality [8], [9]. The major difference between QFD and other quality methodologies resides in a fact that unlike traditional quality systems, which aim at minimizing negative quality in a product, QFD adds values to a product by means of maximizing its positive qualities [10]. The most powerful tool of QFD is House of Quality (HoQ) [11]. It is used to link customers' requirements to design attributes. It establishes explicit relationships between customer's requirements and design attributes, so quality goals in terms of satisfaction of customer's requirements can be traced to technical targets of design attributes. Although QFD was applied in software product and process improvements [12], [13], [14], it has not been applied to development and quality management of SOA-based web service systems.

There are at least two challenging issues in the development of SOA-based web service systems. The first is that we need to explicitly link design attributes to quality of service requirements, and the second is to set up technical targets for design attributes based on quality goals in terms of satisfaction of quality of service requirements. The target problem addressed in this paper, therefore, is to set up technical targets of design attributes of web service systems in terms of quality goals and in consistency with relationships

between quality of service requirements and design attributes of a web service system.

In this paper, we develop a framework for technical target setting for designing high quality SOA-based web service systems using QFD. In this framework, we identify a number of critical web service requirements from perspectives of both web system platform development users and application users and their related design attributes, and construct a house of quality for linking them to develop a web service system. It enables the explicit tracing of customer's needs, represented in terms of quality attributes and design attributes related to web services. Technical targets of the design attributes are determined based on the analysis of the impact of design attributes on customer's needs using a Bayesian Regularized Artificial Neural Network (ANN) method. It links what software designers do directly with what customers want in terms of quality of services in SOA-based web service systems.

The rest of the paper is organized as follows: Section 2 describes related works, including a discussion of quality of web service systems, technical target setting, and ANN methods used in QFD. Section 3 presents a framework of our work, including a discussion of architecture of web service systems using QFD and house of quality, the quality of service requirements for SOA, and related technical design attributes. Section 4 presents principals and a method of technical target setting in QFD using Bayesian regularized artificial neural network technique. Section 5 presents an application example using a SOA infrastructure web application platform. Finally, Section 6 draws conclusions.

## 2 Related Works

### 2.1 Quality of Web Service Systems

Service requirements drive design of SOA-based web service systems. Selection and design of architecture for such systems that satisfy functional as well as nonfunctional requirements are vital to their success. SOA is architecture for a system that is built using a set of services. It is important to analyze how SOA-based web service systems support quality requirements. A set of 13 quality attributes have been identified as important factors in the design of SOAs [15]: interoperability, reliability, availability, usability, security, performance, scalability, extensibility, adaptability, testability, auditability, operability and deployability, and modifiability.

Quality of service requirements, represented in terms of these quality attributes, play an important role in the design of SOAs. Unfortunately, there are no systematic methods which explicitly and quantitatively trace quality of service requirements into design attributes. Only a small number of design parameters, such as network bandwidth and download speed, are analyzed in several web service products' technical specifications [5], [6].

### 2.2 Technical Targets Setting in QFD

QFD can be used to explicitly and quantitatively trace quality of service requirements to design attributes. After identifying the relationship between quality of service requirements and design attributes, we should be able to analyze and achieve a quality target through improvements in design attributes. In addition, we can also determine if the subjective quality target is reasonable, or not, based on an analysis of explicit traceability.

Three quantitative approaches for setting technical target values are widely used in QFD: benchmarking, primitive linear regression, and impact-analysis-based linear regression have been developed. They are based on relationships between quality ratings of competitive products in terms of satisfaction of requirements and their technical performance measurements in terms of design attributes [12], [13], [14]. Other regression methods, such as logarithmic, exponential, and quantic are also being mentioned in [14].

The use of linear regression is an improvement over the benchmarking in setting targets, because it takes into consideration the design attribute values of all competitors and their corresponding customer satisfaction ratings, thereby resulting in a better target value for attaining the desired customer satisfaction value. However, in many cases, the relationship between the customer satisfaction level and the design attribute value is not so simple that it can be represented by a linear relationship. In such complex cases, we can use nonlinear regression to represent the relationship between the customer satisfaction and design attribute values. Unfortunately, it is very hard to find a nonlinear regression model in many cases.

### 2.3 Application of ANN in QFD

An artificial neural network, often just called a "Neural Network" (NN), is a computational model based on biological neural networks. It consists of an interconnected group of artificial neurons and processes information using a connectionist approach to computation. In most cases, an ANN is an adaptive system that changes its structure based on external or internal information that flows through the network during the learning phase.

In more practical terms, neural networks are nonlinear statistical data modeling tools. They can be used to model complex relationships between inputs and outputs or to find patterns in data. They are widely applied in automation, pattern recognition, and optimization.

Their applications in QFD can be classified into two categories: 1) they are used to make the QFD matrices more objective [15], [16], [17], [18], and 2) they are used in QFD to optimize quality planning for achieving technical targets [19], [20], [21], [22], [23].

In this paper, we are going to develop a computational model using ANN, which represents relationships between the data sets of quality of service requirements and technical design attributes. It will be used for setting technical targets based on satisfaction of quality of service requirements. In addition, it will be used to assess the impact of underachieved or overachieved technical targets on the satisfaction of quality of service requirements.

## 3 Design of Web Service Systems for Satisfying Customer's Needs Using QFD

The main issue in designing a web service system is to trace the quality of service requirements to technical design attributes. In this section, we will discuss the application of QFD to the design of web service systems, the architecture of the web service system, from which we can map the

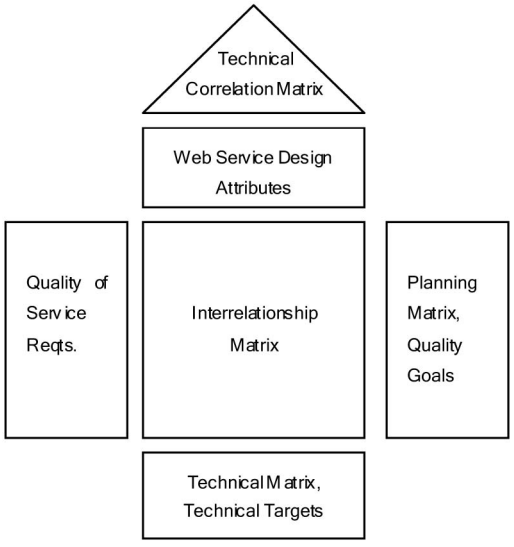



Fig. 1. House of quality in QFD.

nonfunctional customer's requirements in terms of quality of service requirements to technical design attributes.

### 3.1 Application of QFD and House of Quality for Web Service Systems

QFD is a methodology for incorporating the voices of customers, both spoken and unspoken, into development of a product. Nowadays, QFD has been applied in many industries, including software development [12], [13], [14], although it has not been applied in the development of SOA-based web service systems.

An important tool in QFD is the House of Quality (HoQ). A detailed description of HoQ is presented by Hauser and Clausing [11]. Fig. 1 shows a House of Quality for SOA-based web service systems. It connects quality of service requirements for SOA-based web service systems to service design attributes from designers' perspectives. The House of Quality contains six major components:

1. Quality of Service Requirements (WHAT's): A structured list of quality of service requirements derived from customer's needs.
2. Web Service Design Attributes (HOW's): A structured set of relevant and measurable technical service characteristics which affect SOA quality of service requirements.
3. Planning matrix: Identify and analyze customer's perceptions observed in market surveys, including competitor's performance in meeting the SOA quality of service requirements and the quality goal of our product.
4. Interrelationship matrix: Illustrates the QFD team's perceptions of interrelationships between web service design attributes and quality of service requirements. It traces what customers want to what designers can do in the development of SOA-based web service systems. It enables their designers to design them based on traceability analysis. Each entry in the matrix represents the impact of a web service design attribute on a quality of service requirement using an appropriate scale.
5. Technical correlation (Roof) matrix: Used to identify where web service design attributes support or impede each other in the product design.
6. Technical matrix and technical targets: Used to measure the technical performance of competitive products and to specify technical targets our product should reach in order to achieve our goals of quality of service requirements.

The steps in designing a web service system, for satisfying quality of service requirements using QFD, can be described as follows:

1. Identify a set of quality of service requirements, and define a quality goal for each requirement, based on the analysis of competitive products in the planning matrix.
2. Identify a set of important web service design attributes which affect satisfaction with the quality of service requirements, analyze their impacts on the requirements in the interrelationship matrix, and perform technical competitive analysis of competitive products in the technical matrix.
3. Establish the relationships between quality of service requirements and web service design attributes.
4. According to the relationships determined in step 3, trace our quality goals to technical targets which will provide guidance for the design of a web service system.
5. If the technical targets set in step 3 are not achieved in practice, assess the impact of unachieved quality goals.

The approach, which includes step 3 to step 5, is introduced in [13], [14].

Technical target setting usually involves trade-offs among design attributes and customer's requirements. In general, designers of web systems try to walk a fine line between too aggressive technical targets, which require too much resource to implement although they may help to beat competitors, and too conservative technical targets, which usually result in losing in competition even though they may require fewer resources to implement. For example, assume that we plan to develop a web search service. One of the design attributes of a web search service is the number of webpages/weblinks which need to be stored in its search database to address customer's needs for search accuracy, efficiency, and scalability. If too many webpages/weblinks are stored, search efficiency may be decreased and too much storage is required. If too few webpages are stored, search accuracy will be worse than its competitors. A house of quality can be developed for the web search service and a technical target for the size of the web database can be set up accordingly, so that its overall quality in terms of satisfaction with customer's requirements on accuracy, efficiency, and scalability is better than that of its competitors. This provides an example of the use of a house of quality in a simple setting.

### 3.2 Architecture of an SOA-Based Web Application Platform

SOA is important to several different stakeholders who may have different quality of service requirements. To

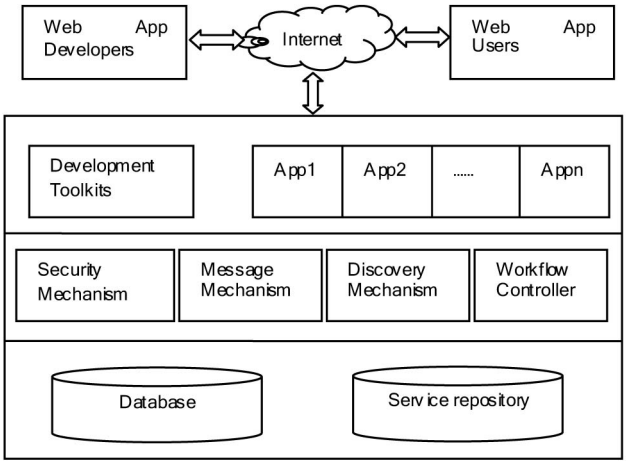



Fig. 2. Architecture of a web application platform.

developers and solution architects, service orientation is a means for creating dynamic and collaborative applications. To the enterprise manager, service orientation is a means for effectively integrating the diverse systems typical of modern enterprise data centers. To the SOA vendors, service orientation is a crucial prerequisite to creating applications that leverage the network to link the actors and systems that drive business processes [24]. From the view point of designing SOA-based products, the quality requirements of all different stakeholders should be taken into consideration.

As we mentioned earlier, SOA does not mean web services, but web services are based on accepted standards and are driving SOA to the mainstream.

We will use a web application platform, which is a hypothetical SOA infrastructure platform, to illustrate the application of QFD to the design of SOA-based web service systems. Web application developers use the platform to develop web applications, and web application users use the applications on the same platform too. Both kinds of users do not need to maintain servers which run the platform and leave the work to platform vendors.

Fig. 2 shows architecture for the platform. Web developers and web application users access the platform through the Internet. Many development toolkits can make the development of a web application easily and effectively. Lots of web applications can meet the needs of all kinds of users and make the interoperation of some kinds of them possible.

In a SOA, services are described in a standard specification language, have a published interface, and communicate with each other requesting execution of their operations in order to collectively support a common business task or process. All of the services are stored in the service repository, and message mechanism is necessary for communication between services. Discovery mechanism is used to search the needed services and multiple text information such as website pages or images.

The most attractive benefit of SOA is its agility [24], and the effective workflow and process management mechanism guarantee it. Otherwise, because of the open environment of the Internet, effective security mechanism is needed, such as integrated Single Sign-On (SSO), access management, directory services, and federated trust across heterogeneous systems, etc.

For enterprise web applications, large amounts of information such as user list, multimedia data, etc., need to be stored and an effective database management system is needed.

TABLE 1
Quality of Service Requirements of a Web Application Platform

| Quality of Service Requirements | Description |
|---|---|
| R1: Interoperability | Refers to the ability of a collection of communicating entities to share specific information and operate on it according to agreed-upon operational semantics. |
| R2: Usability | Ease with which people can employ a particular tool or other human-made object in order to achieve a particular goal. |
| R3: Adaptability | Ability of the system to adapt when users customize their own application functions and interfaces. |
| R4: Performance | Usually related to response time (how long it takes to process a request), throughput (how many requests overall can be processed per unit of time), or timeliness (ability to meet deadlines, i.e., to process a request in a deterministic and acceptable amount of time). |
| R5: Reliability | Ability of a system or component to perform its required functions under stated conditions for a specified period of time. |
| R6: Scalability | Ways for dealing with an increase in the number of service users and the increased need to support more requests for services. |

*All six quality service requirements are adapted from reference [24]*

### 3.3 Quality of Service Requirements and Service Design Attributes of a Web Application Platform

Quality of service requirements are usually specified in terms of quality attributes. Because there are a lot of misconceptions and hype surrounding SOA, there are no relatively mature standards about the quality attributes for SOA-based web service systems [24]. Here we choose six quality attributes from the 13 related quality attributes [24] for quality of service requirements: interoperability, adaptability, usability, performance, reliability, and scalability. The detailed description of these quality attributes can be found in O'Brien et. al. [24], and their summary is described in Table 1.

We identify and analyze 10 typical web service design attributes which affect the quality of service requirements. They are listed in Table 2 below.

- T1: "# of languages" represents how many programming languages the platform can support. The more languages supported, the more choice of languages the developers will have, and the more interoperation tools that are needed so that the platform may have higher interoperability.
- T2: "# of data types" represents how many original data types the platform supports. The more data types supported, the easier programming is.

TABLE 2
Web Service Design Attributes of a Web Application Platform

| Web Service Design Attributes | Description |
|---|---|
| T1: # of Languages | Number of programming languages used in the platform. |
| T2: # of Data types | Web application development needs data types: ordinary data, media data and text etc. |
| T3: # of Data Access Protocols | Protocols provided for services. Most common protocols are Http, Https, BitTorrent etc. |
| T4: # of Services | Number of services provided for quick and easy website development. |
| T5: Database Capacity (Billion Objects) | Objects can be stored in the database. |
| T6: Downtime Per Week (Mins) | Average downtime per week indicates the probability of web application in status of downtime. |
| T7: Bandwidth(MB/s) | Capacity of a given system to transfer data over a connection. |
| T8: Maximum users | Maximum number of users the web application supports |
| T9: Query Response Time(s) | Average query response time is the performance bottleneck for dynamic web pages. |
| T10: Download Speed(MB/s) | Average normal download speed. |

- T3: "# of data access protocols" represents the number of protocols provided for services. Most common protocols are Http, Https, BitTorrent, etc. The more protocols it has, the easier the developers develop their applications. The more protocols, the easier the users find the applications they need.
- T4: "# of services" is the number of services the platform provided. The higher the number of services, the more easily the user's needs are met.
- T5: "database capacity" represents the size of the database. The database is the core of the data centers in the platform. The bigger the database size, the better the scalability, the adaptability, and the usability of the system.
- T6: "downtime per week" indicates the probability of web application in the status of the downtime. Since downtime cost is huge for a big enterprise, system stability is necessary for a successful SOA system. If a system's availability is 99 percent, then its downtime per week is 1.68 hours. And if a system's availability is 99.9999 percent, then its downtime per week is 0.605 seconds. This design attribute affects a system's performance and reliability.
- T7: "Bandwidth" is the capacity for the system to transfer data on the Internet. It is certainly an important measurement for a system's performance.
- T8: "Maximum users" represents how many users can use the application. The bigger the value, the higher the system's scalability.

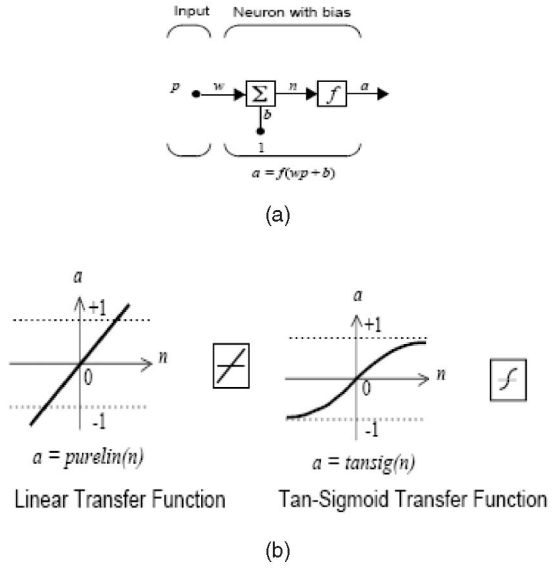



Fig. 3. Simple neuron and its transfer functions [25]. (a) Simple neuron with bias. (b) Two transfer functions.

- T9: "Query response time" is the average response time for a system to perform a query. Since most webpages are dynamically created in an application system, it is usually a performance bottleneck for web applications.
- T10: "Download speed" is the average normal download speed for a system, and it affects its performance.

## 4 Neural Networks

An artificial network consists of a pool of simple processing units which communicate by sending signals to each other over a large number of weighted connections. Artificial neural network models are essentially simple mathematical models defining a function $f : P \rightarrow A$. Each type of ANN model corresponds to a class of such functions [25].

### 4.1 Simple Neuron and Its Transfer Function

Fig. 3 illustrates a simple neuron and its transfer functions used in our neural network [25]. The scalar input $p$ is transmitted through a connection that multiplies its strength by scalar weight $w$, to form the product $wp$, again a scalar. You may view scalar bias, $b$, as simply being added to the product wp as shown by the summing function or as shifting the function f to the left by an amount $b$. The bias is much like a weight, except that it has a constant input of 1. The transfer function net input n, again a scalar, is the sum of the weighted input wp and the bias $b$. This sum is the argument of the transfer function f which is typically a sigmoid function, which takes the argument n and produces the output a. Note that w and b are both adjustable scalar parameters of the neuron. These parameters can be adjusted so that the network exhibits desired or interesting behavior, and it can be trained to do a particular job. The linear transfer function and the sigmoid transfer function are also shown in Fig. 3. The former calculates the neuron's output by simply returning the value passed to it. The latter takes the input, which may

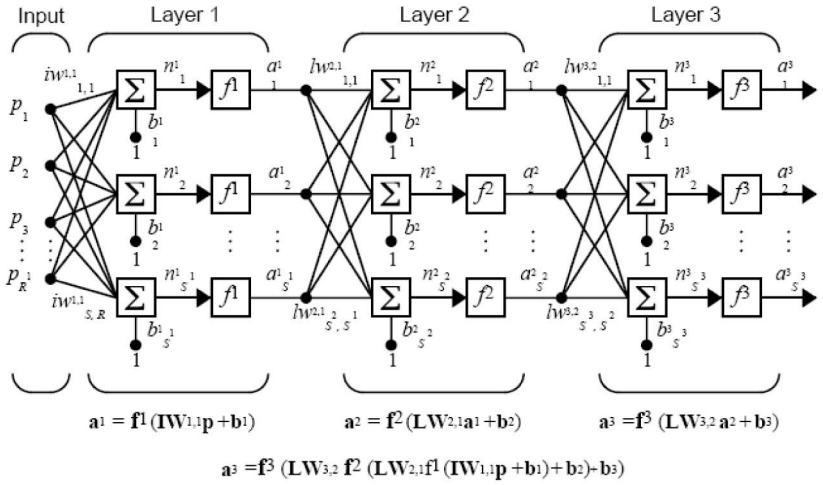



Fig. 4. Three-layer artificial neural network [25].

have any value between plus and minus infinity, and squashes the output into the range 0 to 1.

### 4.2 Multiple-Layer Neural Network

A multiple-layer neural network can be considered as a map from inputs to outputs; each layer has its own weight matrix, bias, and transfer function. An example of multiple-layer neural network of three layers is shown in Fig. 4, where the first layer is sigmoid and the second layer is linear. It can be trained to approximate any function (with a finite number of discontinuities) [26].

The network shown above has $R^1$ inputs, $S^1$ neurons in the first layer, $S^2$ neurons in the second layer, etc. It is common for different layers to have different numbers of neurons. A constant input 1 is fed to the biases for each neuron. Note that the outputs of each intermediate layer are the inputs to the following layer. Thus, the second layer can be analyzed as a one-layer network with $S^1$ inputs, $S^2$ neurons, and an $S^2 \times S^1$ weight matrix $W^2$. The input to the second layer is $a^1$; the output is $a^2$. Now that we have identified all the vectors and matrices of the second layer, we can treat it as a single-layer network on its own. This approach can be taken with any layer of the network.

Here we assume that the output of the third layer, $\mathbf{a}^3$, is the network output of interest. In this three-layer ANN,

$$\mathbf{a}^1 = \mathbf{f}^1(\mathbf{IW}^{1,1}\mathbf{p} + \mathbf{b}^1), \quad (1)$$

$$\mathbf{a}^2 = \mathbf{f}^2(\mathbf{LW}^{2,1}\mathbf{a}^1 + \mathbf{b}^2), \quad (2)$$

$$\mathbf{a}^3 = \mathbf{f}^3(\mathbf{LW}^{3,2}\mathbf{a}^2 + \mathbf{b}^3). \quad (3)$$

$a^3$ can also be expressed as:

$$\mathbf{a}^3 = \mathbf{f}^3(\mathbf{LW}^{3,2}\mathbf{f}^2(\mathbf{LW}^{2,1}(\mathbf{IW}^{1,1}\mathbf{p} = \mathbf{b}^1) + \mathbf{b}^2) + \mathbf{b}^3). \quad (4)$$

It represents the output of the multilayer network.

### 4.3 The Training Process of the Neural Network

The greatest advantage of ANN is that it has a learning ability to adapt its network to a structure. If a transfer function is given, then the main task of learning is obviously focused on the weight matrix and bias of each layer.

Training a neural network model essentially means selecting one model from the set of allowed models that minimizes the cost criterion. There are numerous algorithms available for training neural network models, and most of them can be viewed as a straightforward application of optimization theory and statistical estimation.

Most of the algorithms used in training artificial neural networks use some forms of gradient descent. This is done by simply taking the derivative of the cost function with respect to the network parameters and then changing those parameters in a gradient-related direction.

A commonly used cost is the mean-squared error which tries to minimize the average squared error between the network's output, *f(p)*, and the target value $a$ over all of the exemplary pairs. When we try to minimize this cost using gradient descent for the class of neural networks, we obtain common and well-known backpropagation algorithm for training neural networks.

Matlab provides a convenient way to construct an ANN structure, to train the network and get a layer weight matrix or bias, and to simulate an ANN in many different ways. If we use the customer's requirements data set and design attributes set in a house of quality as inputs and outputs, respectively, we can train an appropriate network and predict the technical targets. On the other hand, if we use the design attributes set as inputs and the customer's requirements data set as outputs, we can train an appropriate network and evaluate the customer's requirements satisfaction levels in terms of design attribute ratings, which will help assess the impact of underachieved or overachieved technical targets in satisfaction of customer's requirements and quality goal.

Bayesian regularization is an effective training method [27], [28], [29], [30], [31], [32], [33], [34]. It updates the weight and bias values according to Levenberg-Marquardt optimization, minimizes a combination of squared errors and weights and then determines the correct combination to produce a network that generalizes well. Our experiment showed that Bayesian regularization algorithm had satisfactory training speed and accuracy throughout the tests. Unlike a standard feedforward neural network training method, where a single set of parameters (weights, biases, etc.) are used, the Bayesian approach to neural network modeling considers all possible values of network parameters,

weighted by the probability of each set of weights. The proposed Bayesian regularized back-propagation neural network is used to overcome the linear regression method's deficiencies and investigate nonlinearity.

### 4.4 Process for Training the Bayesian Regularized ANN in Matlab

Matlab provides a powerful toolbox for training a Bayesian Regularized ANN. The training process can be described as follows:

Step 1. Prepare the inputs and outputs data sets, P and T, respectively.

The customer's requirement satisfaction data set and design attribute evaluation set in the house of quality can be used as inputs/outputs or outputs/inputs according to the purpose of predicting technical targets or customer's requirements satisfaction levels, respectively.

Step 2. Normalize the corresponding data sets P and T.

In order to relieve the training difficulty and provide a balance of parameters during the training process, the data sets P and T are normalized. It is recommended that the data be normalized between a lower bound and upper bound, such as 0.1 and 0.9. One way to scale input and output variables in the interval [0.1, 0.9] is as follows:

$$\begin{aligned}\mathbf{P_n} &= \mathbf{0.1} + (\mathbf{0.9} - \mathbf{0.1}) \times (\mathbf{P} - \mathbf{P_{min}})/(\mathbf{P_{max}} - \mathbf{P_{min}}),\\ \mathbf{T_n} &= \mathbf{0.1} + (\mathbf{0.9} - \mathbf{0.1}) \times (\mathbf{T} - \mathbf{T_{min}})/(\mathbf{T_{max}} - \mathbf{T_{min}}),\end{aligned} \quad (5)$$

where $\mathbf{P_n}$, $\mathbf{T_n}$ are the normalized values of $\mathbf{P}$ and $\mathbf{T}$, respectively, and $P_{max}$, $T_{max}$ and $P_{min}$, $T_{min}$ are the maximum and minimum values of $\mathbf{P}$ and $\mathbf{T}$, respectively.

Step 3. Construct the ANN.

In this research, we use a forward-feed ANN.

The function newff creates a feedforward network. It requires four inputs and returns an ANN object. The first input is an $R \times 2$ matrix of minimum and maximum values for each of the $\mathbf{R}$ elements of the input vector. The second input is an array containing the sizes of each layer. The third input is a cell array containing the names of the transfer functions to be used in each layer. The final input contains the name of the training function to be used. For example, the following command creates a three-layer network:

$$\begin{aligned}\mathbf{net} = \mathbf{newff}(&\mathbf{minmax}(\mathbf{P}), [\mathbf{6}, \mathbf{8}, \mathbf{10}],\\ &\{\textbf{'tansig'}, \textbf{'tansig'}, \textbf{'purelin'}\}, \textbf{'trainbr'}).\end{aligned}$$

The three-layer network has one input vector $\mathbf{P}$, six neurons in the first layer, eight neurons in the second layer, and 10 neurons in the third (ouput) layer. Both the transfer functions of the first and second layer are tan-sigmoid represented by "*tansig*" and the output layer transfer function is linear represented by "*purelin*" in the command. These functions are discussed in Fig. 3. Its training function is *trainbr* denoting Bayesian regularization training method. This command creates an ANN object and also initializes the weights and biases of the network.

Step 4. Set up the training parameters.

At this point, we might want to modify some of the default training parameters. When *trainbr* is used, it is important to run the algorithm until parameters are converged. The training may stop with the message "Maximum MU reached." This is typical, and is a good indication that the algorithm has truly converged. It is also converged if the sum squared error (SSE) and sum squared weights (SSW) are relatively constant over several iterations.

There are several important training parameters associated with trainbr: *epochs, goal, and mu_max*.

*epochs*: Maximum number of epochs to train

*goal*: Performance goal

*mu_max*: Maximum value for Marquardt adjustment parameter.

In this implementation, they are set as follows: $epochs = 10{,}000$, $goal = 1e-3$, *and* $mu\_max = 1e100$.

Step 5. Train the ANN.

Now we are ready to train the network. Training stops when any one of following conditions occur:

- The maximum number of *epochs* (repetitions) is reached.
- The maximum amount of time is exceeded.
- Performance is minimized to the *goal*.
- The performance gradient falls below mingrad.
- mu exceeds *mu_max*.
- Validation performance has improved more than *max_fail* times since the last time it reduced in validation.

The train function is used as follows:

$$[\mathbf{net}, \mathbf{tr}] = \mathbf{train}(\mathbf{net}, \mathbf{P}, \mathbf{T}).$$

Step 6. Simulate ANN and unnormalized.

After the neural network was trained, tested, and simulated, it is necessary for the simulating data to be unnormalized. The unnormalized method is as follows:

$$T = (T_n - 0.1) \times (T_{\max} - T_{\min})/(0.9 - 0.1) + T_{\min}, \quad (6)$$

where $T$ is the unnormalized value of $T_n$.

## 5 An Application Example: Design of a Web Application Platform Using QFD

### 5.1 Technical Target Setting

Fig. 5 shows a House of Quality developed for design of the Web Application Platform discussed in Section 3.2, using the process discussed in Section 3.1. It will be used to illustrate technical target setting methods in subsequent sections. It should be noted that data of the web application platform in the house of quality are derived based on the analysis of existing products, for illustration purposes. In practice, they should be collected through rigorous metrics programs and competitive analysis in the development and production process of a web system. Without correct input data in the house of quality for the web system, the target setting methods discussed below cannot produce accurate technical targets like any existing ones [6].

#### 5.1.1 Linear Regression Method for Technical Target Setting

We developed an impact-analysis-based technical target setting method in QFD using linear regression [1], [4]. The relationship between web service design attributes and weighted quality ratings of satisfactions of quality of web

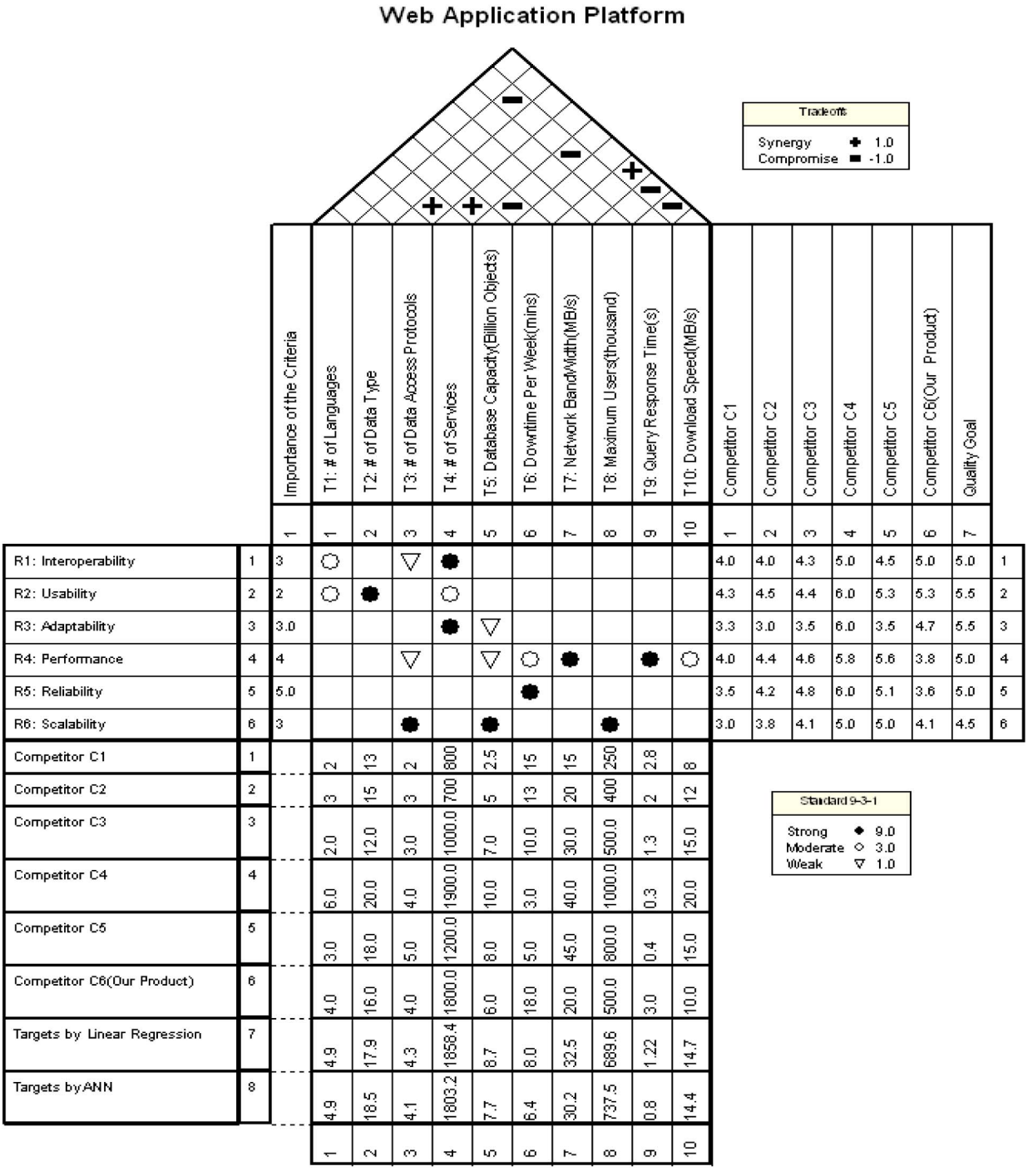



Fig. 5. HoQ of a web application platform.

service requirements in this linear regression method can be represented as follows:

$$y = mx + c, \tag{7}$$

where $y$ represents web service design attribute, and $x$ represents the weighted quality of service requirement. The quality of service requirements are multiplied by their respective impact ratios to obtain their weighted quality of service requirements. The impact ratios can be obtained from the interrelationship matrix in the house of quality in Fig. 5 and the results are shown in Table 3. An element of the matrix represents a contribution ratio of a corresponding design attribute to a corresponding quality requirement. Take R3 as an example, 90 percent of the contribution comes from design attribute T4, and 10 percent from T5.

Table 4 contains regression models of technical targets for design attributes in terms of customers' requirements and their computed technical targets for the application example. If more than one quality of service requirements are influenced by a design attribute, the most aggressive target value is selected. Take T1 as an example. T1 affects R1 and R2. The linear regression model between T1 and R1 is

TABLE 3
Impact Ratio Matrix of a Web Application Platform

| | T1 | T2 | T3 | T4 | T5 | T6 | T7 | T8 | T9 | T10 |
|---|---|---|---|---|---|---|---|---|---|---|
| R1 | 0.23 | | 0.08 | 0.69 | | | | | | |
| R2 | 0.20 | 0.60 | | 0.20 | | | | | | |
| R3 | | | | 0.90 | 0.10 | | | | | |
| R4 | | | 0.038 | 0.038 | | 0.115 | 0.346 | | 0.346 | 0.115 |
| R5 | | | | | | 1.00 | | | | |
| R6 | | | 0.33 | | 0.33 | | | 0.33 | | |

$T1 = 3.5806^{*}R1 - 12.994$. In order to achieve a quality goal of service requirement R1 $QG(R1) = 5.0$, as shown in Fig. 5, T1 should be 4.90. Similarly, the relationship between T1 and R2 is $T1 = 2.5513^{*}R2 - 9.6716$, in order to achieve a quality goal of service requirement R2 $QG(R2) = 5.5$, as shown in Fig. 5, T1 should be 4.36. Since we need to achieve quality goals of both R1 and R2, we should choose the most aggressive one, which means that T1 should be 4.9. For detailed discussion of the impact-based technical target setting, using linear regression, please refer to [1], [4].

### 5.1.2 Bayesian Regularized Neural Network Method

As we described in Section 4, the main issue with ANN is problem of training the network and getting the appropriate weight matrix W and bias b. Matlab neural toolbox provides many functions for creating and training an ANN. We chose the well-known backpropagation algorithm to construct our ANN for technical target setting in QFD, for design of the web application platform. We get the input and output of the ANN from the house of quality in Fig. 5, as below:

$$\begin{aligned} P = [&4.0 \quad 4.0 \quad 4.3 \quad 5.0 \quad 4.5 \quad 5.0; \\ &4.3 \quad 4.5 \quad 4.4 \quad 6.0 \quad 5.3 \quad 5.3; \\ &3.3 \quad 3.0 \quad 3.5 \quad 6.0 \quad 3.5 \quad 4.7; \\ &4.0 \quad 4.4 \quad 4.6 \quad 5.8 \quad 5.6 \quad 3.8; \\ &3.5 \quad 4.2 \quad 4.8 \quad 6.0 \quad 5.1 \quad 3.6; \\ &3.0 \quad 3.8 \quad 4.1 \quad 5.0 \quad 5.0 \quad 4.1] \end{aligned}$$

is used as input. It is a $6 \times 6$ matrix, from six customer requirements and the competitive quality evaluation data from six competitors.

$$\begin{aligned} T = [&2.0 \quad 3.0 \quad 2.0 \quad 6.0 \quad 3.0 \quad 4.0; \\ &13.0 \quad 15.0 \quad 12.0 \quad 20.0 \quad 18.0 \quad 16.0; \\ &2.0 \quad 3.0 \quad 3.0 \quad 4.0 \quad 5.0 \quad 4.0; \\ &800 \quad 700 \quad 1{,}000 \quad 1{,}900 \quad 1{,}200 \quad 1{,}800; \\ &2.5 \quad 5.0 \quad 7.0 \quad 10.0 \quad 8.0 \quad 6.0; \\ &15.0 \quad 13.0 \quad 10.0 \quad 3.0 \quad 5.0 \quad 18.0; \\ &15.0 \quad 20.0 \quad 30.0 \quad 40.0 \quad 45.0 \quad 20.0; \\ &250 \quad 400 \quad 500 \quad 1{,}000 \quad 800 \quad 500; \\ &2.8 \quad 2.0 \quad 1.3 \quad 0.3 \quad 0.4 \quad 3.0; \\ &8.0 \quad 12.0 \quad 15.0 \quad 20.0 \quad 15.0 \quad 10.0] \end{aligned}$$

TABLE 4
The Linear Regression Solution of all Web Service Design Attributes

| | Equations | Targets | Aggr. Tgt. | Fea. ? |
|---|---|---|---|---|
| T1: # of Languages | T1=3.5806*R1-12.994 | 4.909 | 4.9 | No |
| | T1=2.5513*R2-9.6716 | 4.36055 | | |
| T2: # of Data Types | T2=4.1935*R2-5.1613 | 17.90295 | 17.9 | No |
| T3: # of Data Access Protocols | T3=1.5484*R1-3.4161 | 4.3259 | 4.3 | No |
| | T3=0.7895*R4-0.2105 | 3.737 | | |
| | T3=1.2442*R6-1.6843 | 4.3156 | | |
| T4: # of Services | T4=1109.7*R1-3723.2 | 1825.3 | 1858 | No |
| | T4=675.95*R2-2123.9 | 1593.825 | | |
| | T4=416.67*R3-433.33 | 1858.355 | | |
| T5: Database Capacity (Bill.Obj.) | T5=1.5355*R3+0.275 | 8.71995 | 8.7 | No |
| | T5=2.5877*R4-5.7456 | 7.1894 | | |
| | T5=3.2431*R6-7.0962 | 7.49775 | | |
| T6: Downtime Per Week(mins) | T6=-6.959*R4+43.374 | 8.5785 | 7.95 | No |
| | T6=-5.82*R5+37.051 | 7.951 | | |
| T7: Network BandWidth (MB/s) | T7=13.743*R4-36.257 | 32.458 | 32 | No |
| T8: Maximum Users(ths.) | T8=343.89*R6-857.89 | 689.615 | 690 | Yes |
| T9: Query Response Time(s) | T9=-1.3743*R4+8.092 | 1.2209 | 1.22 | No |
| T10: Download Speed (MB/s) | T10=4.6199*R4-8.380 | 14.7194 | 14.7 | No |

*The column "Feasible" indicates if the technical target is feasible or not according our company's technological and financial limitations.*

is used as output. It is a $10 \times 6$ matrix, from 10 design attributes and the competitive technical evaluation data from six competitors.

A three-layer backpropagation neural network with a number of layer nodes: "6-8-10", and layer transfer functions: "tansig-tansig-purelin" is used. In order to get the best results, the training data are first mapped into a range $[-1, 1]$. We scale both input P and output T accordingly. Because the original data are at a different order of magnitude, we used (5) to normalize the data, to map the data to a range [0, 1].

After its performance goal is reached, as shown in Fig. 6, the trained network has the weight matrices and bias, as following:

$$\mathbf{IW}^{1,1} =$$
$$\begin{bmatrix} 0.0602 & -0.5044 & -0.2300 & 0.0859 & 0.3187 & 0.2547; \\ 0.0886 & 0.0592 & 0.1211 & 0.0506 & 0.0468 & 0.0523; \\ -0.2563 & -0.1082 & 0.2256 & -0.2222 & -0.1120 & 0.7255; \\ -0.0886 & -0.0592 & -0.1211 & -0.0506 & -0.0468 & -0.0523; \\ 0.0886 & 0.0592 & 0.1211 & 0.0506 & 0.0468 & 0.0523; \\ 0.6795 & 0.2197 & -0.3166 & -0.5277 & 0.1051 & -0.2892 \end{bmatrix},$$

$$\mathbf{b}^1 = [-0.0533 \quad 0.0074 \quad -0.3048 \quad -0.0074 \quad 0.0074 \quad 0.4055]',$$

$$\mathbf{LW}^{2,1} =$$
$$\begin{bmatrix} -0.0599 & 0.2753 & 0.1538 & -0.2753 & 0.2753 & -0.0172; \\ -0.0599 & 0.2753 & 0.1538 & -0.2753 & 0.2753 & -0.0172; \\ 0.0599 & -0.2753 & -0.1538 & 0.2753 & -0.2753 & 0.0172; \\ -0.2416 & 0.0468 & -0.3606 & -0.0468 & 0.0468 & 1.0851; \\ -0.0599 & 0.2753 & 0.1538 & -0.2753 & 0.2753 & -0.0172; \\ 0.1383 & 0.0840 & 0.8705 & -0.0840 & 0.0840 & -0.0787; \\ 0.6500 & 0.0062 & -0.1168 & -0.0062 & 0.0062 & 0.2860; \\ -0.0599 & 0.2753 & 0.1538 & -0.2753 & 0.2753 & -0.0172 \end{bmatrix},$$

$$\mathbf{b}^2 = [0.2778 \quad 0.2778 \quad -0.2778 \quad 0.1187 \quad 0.2778 \quad 0.0396 \quad 0.0115 \quad 0.2778]',$$

$$\mathrm{LW}^{3,2} =$$
$$\begin{bmatrix} 0.0585 & 0.0585 & -0.0585 & -0.0371 & 0.0585 & 0.2817 & -0.5411 & 0.0585; \\ 0.2197 & 0.2197 & -0.2197 & -0.0709 & 0.2197 & -0.0029 & -0.1409 & 0.2197; \\ 0.0675 & 0.0675 & -0.0675 & 0.0003 & 0.0675 & 0.4865 & -0.0046 & 0.0675; \\ 0.5515 & 0.5515 & -0.5515 & 0.0636 & 0.5515 & 0.2894 & -0.3399 & 0.5515; \\ 0.1037 & 0.1037 & -0.1037 & -0.0967 & 0.1037 & 0.6534 & 0.0475 & 0.1037; \\ 0.1492 & 0.1492 & -0.1492 & 0.6430 & 0.1492 & 0.1304 & 0.2865 & 0.1492; \\ 0.2674 & 0.2674 & -0.2674 & -0.2613 & 0.2674 & 0.2002 & 0.1870 & 0.2674; \\ 0.4919 & 0.4919 & -0.4919 & -0.1941 & 0.4919 & 0.3242 & -0.0132 & 0.4919; \\ -0.0235 & -0.0235 & 0.0235 & 0.8663 & -0.0235 & 0.0559 & 0.1467 & -0.0235; \\ 0.2025 & 0.2025 & -0.2025 & -0.1922 & 0.2025 & 0.1336 & 0.1000 & 0.2025 \end{bmatrix},$$

$$\mathbf{b}^3 = [0.0482 \quad 0.2943 \quad 0.0254 \quad 0.6323 \quad 0.0721 \quad 0.2049 \quad 0.3231 \quad 0.6126 \quad 0.0110 \quad 0.2773]'.$$

In the above bias, $[\,]'$ represents the transpose of a matrix.

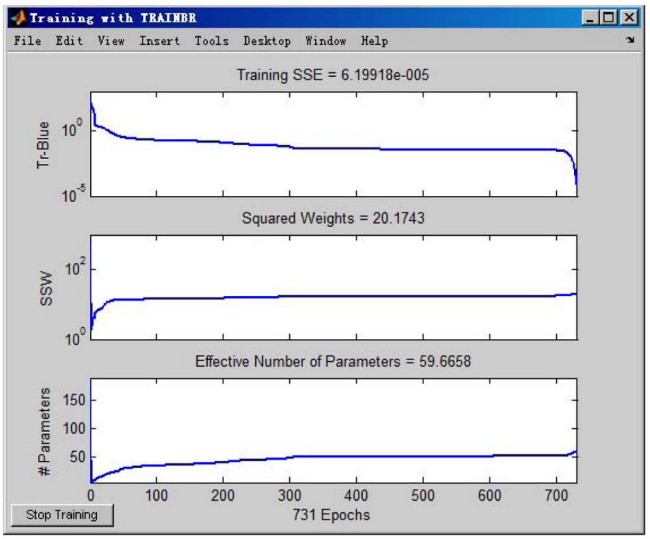



Fig. 6. Training process of ANN for technical target setting. TRAINBR-calcjx, Epoch 731/10000, SSE 6.19918e-005/0.0001, SSW 20.1743, Grad 2.58e-002/1.00e-010, #Par 5.97e+001/188 TRAINBR, Performance goal met. MSE = 1.0332e-006.

TABLE 5
The Technical Targets Comparison
Between Linear Regression and ANN Methods

| The customer's Reqts. Goal | T1: # of Languages | T2: # of Data Types | T3: # of Data Access Protocols | T4: # of Services | T5: Database Capacity (Billion Objects) | T6: Downtime Per Week (mins) | T7: Network BandWidth (MB/s) | T8: Maximum Users (thousand) | T9: Query Response Time(s) | T10: Download Speed (MB/s) |
|---|---|---|---|---|---|---|---|---|---|---|
| R1: 6 | | | | | | | | | | |
| R2: 6 | | | | | | | | | | |
| R3: 5.5 | | | | | | | | | | |
| R4: 5 | | | | | | | | | | |
| R5: 5 | | | | | | | | | | |
| R6: 5 | | | | | | | | | | |
| Linear Regression | 4.9 | 18 | 4.3 | 18 | 8.7 | 8 | 32 | 69 | 1.2 | 15 |
| Bayesian ANN | 4.9 | 19 | 4.1 | 18 | 7.7 | 6.4 | 30 | 76 | 0.8 | 14 |
| Difference (%) | 0 | -3 | 6 | 3 | 13 | 24 | 7 | -6 | 53 | 2 |

*The difference=(Linear Regression value –ANN value)/ANN Value*

Suppose that the expected quality goals of service requirements for satisfaction of six web service requirements are $\mathrm{x} = [5.0 \quad 5.0 \quad 5.5 \quad 5.0 \quad 5.0 \quad 4.5]'$, then technical targets of the ten design attributes are derived from (1)-(4) as follows:

$$\mathbf{a} = [4.9 \quad 18.5 \quad 4.1 \quad 1803.2 \quad 7.7 \quad 6.4 \quad 30.2 \quad 737.5 \quad 0.8 \quad 14.4]'.$$

These results are different from those derived from other methods, such as primitive linear regression which is listed in Table 4. Their difference is listed in Table 5.

Technical target values for all of the web service design attributes $Tj(j = 1, 2, \ldots, 10)$ are computed in Table 5, using the linear regression method.

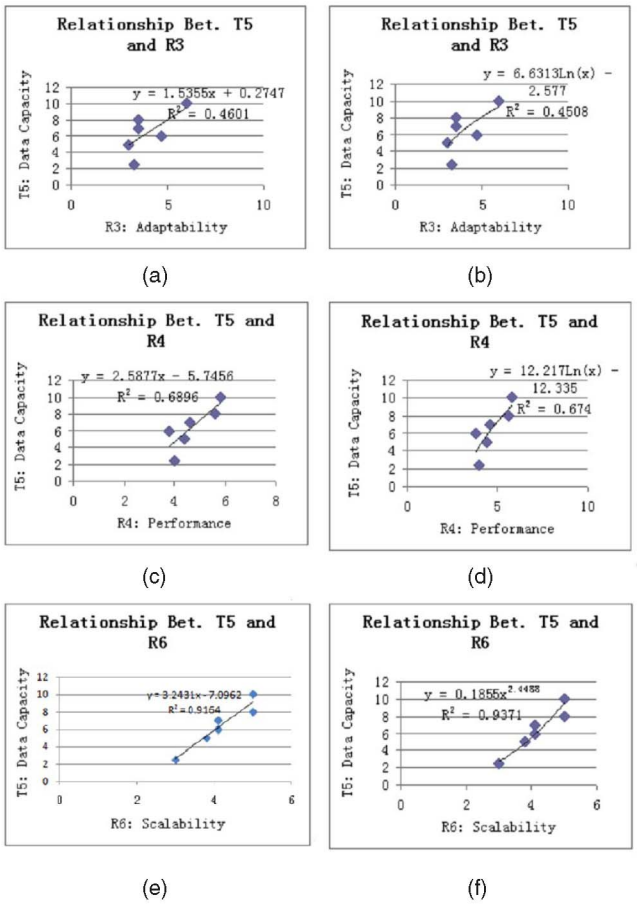



Fig. 7. Scattered point graph to illustrate the relationship between T5 and R3, R4 and R6. (a), (c), and (e) show the scattered point and its linear trend line. (b), (d), (f) present the most suitable trend line except the linear trend, because of the biggest value of $R^2$, which is the correlation coefficient, and when it is bigger, the relationship between the two variables is better.

We classified these technical design attributes in two categories, based on comparison, in Table 5:

1. The Bayesian ANN target is close to the linear regression target. In this case, we think there is no need to recognize which method has the better targets.
2. The Bayesian ANN target and linear regression target are much different from each other. For example, the technical targets of T5, T6 and T9, in both methods, are over 10 percent different.

Please note that in the linear regression method, if a technical design attribute has relationships with more than one customer requirement, we always choose the aggressive one to make sure quality goals of all the customer requirements can be reached. Table 4 illustrates this process.

Fig. 7 shows the scattered data graphs of relationships between T5 and R3, T5 and R4, and T5 and R6.

From the computational process of the technical target, based on the linear regression in Table 4, we can see if the expected quality goal of service requirements $\mathrm{QG(R3)} = 5.5$, $\mathrm{QG(R4)} = 5$, $\mathrm{QG(R6)} = 4.5$, technical target of T5 is set to be $\mathrm{T5} = 8.72$, which is calculated according the relationship between T5 and R3. But we can find that the relationship between T5 and R3 is not linear. When R4 reaches the expected value 4.5, the T5 should set to 7.2 and 7.3, according to the linear relationship equation and logarithmic relationship equation between T5 and R4. When R6 reaches the expected value 4.5, the T5 should set to 7.5 and 7.4, according to the linear relationship equation and power relationship equation between T5 and R6. So from this point, the Bayesian ANN target $\mathrm{T5} = 7.7$ is a better target value than the linear regression target $\mathrm{T5} = 8.72$.

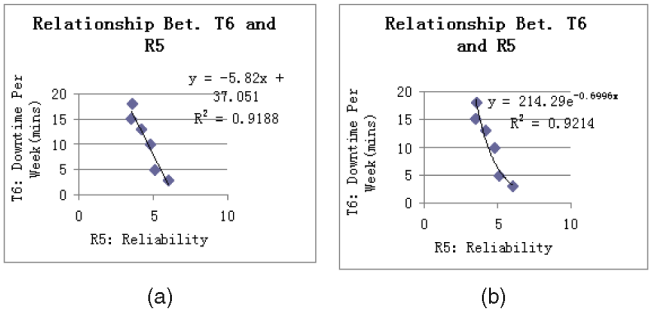



Fig. 8. The relationship between T6 and R5. The trend line is (a) linear and (b) exponential.

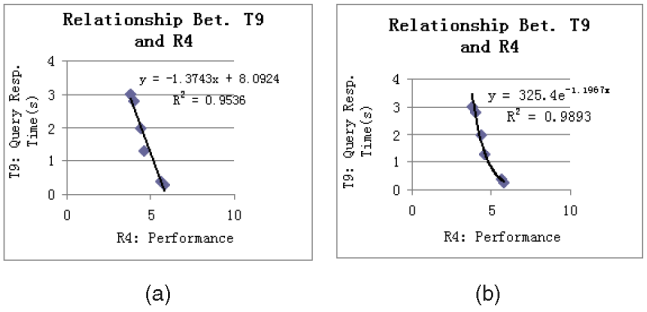



Fig. 9. The relationship between T9 and R4. The trend line is (a) linear and (b) exponential.

Fig. 8 shows the relationship between T6 and R5, which is the most aggressive one in determination of the technical target of T6. From the graph, we find that the relationship between T6 and R5 is exponential instead of linear. According to the exponential relationship equation in Fig. 8b, the predicted target value is 6.47, which is very close to the Bayesian ANN target $\mathrm{T6} = 6.4$. In terms of the target of the technical design attribute T6, the Bayesian ANN method has a more objective target than linear regression does, based on their nonlinear relationship.

Fig. 9 shows the relationship between T9 and R4. From the graph, we find that the relationship between T9 and R4 is exponential instead of linear. According to the exponential relationship equation in Fig. 9b, the predicted target value is 0.82, which is very close to the Bayesian ANN based target $\mathrm{T9} = 0.8$. In terms of the target of technical design attribute T9, Bayesian ANN method has a more objective target than linear regression does, based on their nonlinear relationship.

A two-layer ANN has the ability to approximate any function with arbitrary accuracy [34]. No matter whether the relationship between technical design attribute and the quality of service requirement,is linear or not, the ANN-based model can represent the relationship with high accuracy. Therefore, the ANN-based technical target setting

method can generate more objective targets in terms of both linear and nonlinear relationships between requirements and design attributes.

### 5.2 Assessment of Underachieved and Overachieved Technical Targets

If targets of design attributes are underachieved or overachieved, what level of satisfaction of the quality of service requirements can be achieved?

Since it still can be considered as the relationship between two data sets, ANN can also be used to assess the impact of underachieved or overachieved technical targets on quality level in terms of customer requirement satisfaction.

We just need to exchange the outputs and inputs in the previous ANN, make adjustments to their neuron numbers in each layer, and we can obtain an estimate of customer's satisfaction levels based on the achieved technical levels. The parameters P and T for the ANN for assessment of quality level in terms of satisfaction of customer requirements, based on the achieved technical levels, are the original T and P matrices in Section 5.1, respectively.

We used a back-propagation neural network with three layers, number of layer nodes "10-8-6", and layer transfer functions "tansig-tansig-purelin."

After its performance goal was reached, as shown in Fig. 10, the trained network had the weight matrices and bias as follows:

$$\mathbf{IW^{1,1}} = \begin{bmatrix} -0.2018 & -0.3802 & -0.0125 & -0.4674 & 0.1834 & 0.1364 \\ -0.1010 & -0.1661 & -0.0821 & 0.2440; & & \\ 0.0728 & 0.2677 & 0.0649 & 0.3918 & -0.1085 & -0.0452 \\ 0.1847 & 0.1601 & 0.1283 & -0.1724; & & \\ 0.0307 & -0.0522 & 0.1025 & 0.0623 & -0.1431 & 0.4575 \\ -0.3776 & -0.2401 & 0.4166 & -0.3025; & & \\ -0.5820 & -0.2162 & 0.5158 & -0.5265 & 0.0950 & 0.1355 \\ 0.3127 & -0.0986 & -0.0949 & -0.0430; & & \\ -0.1116 & 0.4914 & 0.6692 & -0.9424 & -0.1195 & 0.0605 \\ 0.0420 & -0.1127 & -0.1530 & -0.2004; & & \\ -0.0224 & 0.0757 & 0.0517 & 0.3201 & 0.0489 & 0.0862 \\ 0.2411 & 0.1573 & 0.1898 & 0.0168; & & \\ -0.2027 & -0.1226 & -0.4242 & -0.2538 & -0.4541 & 0.1251 \\ -0.2798 & -0.2691 & 0.3038 & -0.3575; & & \\ -0.1088 & -0.3294 & -0.0726 & -0.3885 & 0.1445 & 0.0720 \\ -0.1638 & -0.1602 & -0.1145 & 0.2116; & & \\ -0.0390 & -0.2076 & -0.0574 & -0.3924 & 0.0707 & 0.0176 \\ -0.2053 & -0.1606 & -0.1422 & 0.1304; & & \\ -0.0649 & 0.1345 & -0.4742 & -0.4715 & -0.0180 & -0.3823 \\ 0.3772 & 0.2174 & -0.1956 & 0.3959], & & \end{bmatrix}$$

$$\mathbf{b^1} = [0.0301 \quad 0.1179 \quad -0.0794 \quad 0.1353 \quad -0.0293 \quad 0.3227 \quad 0.0367 \quad -0.0746 \quad -0.1636 \quad 0.5714]',$$

$$\mathbf{LW^{2,1}} = \begin{bmatrix} 0.6923 & -0.4787 & -0.0252 & 0.5029 & -0.2711 & -0.1372 \\ -0.2994 & 0.5666 & 0.3904 & 0.0657; & & \\ 0.1746 & -0.2771 & -0.0123 & 0.0168 & 0.1911 & -0.4045 \\ 0.1439 & 0.2485 & 0.3066 & -0.1876; & & \\ -0.1724 & 0.2757 & 0.0176 & -0.0168 & -0.1819 & 0.4047 \\ -0.1486 & -0.2473 & -0.3052 & 0.1887; & & \\ -0.3111 & 0.3260 & -0.3996 & -0.0105 & -0.6658 & 0.3083 \\ 0.0963 & -0.3040 & -0.3443 & 0.0915; & & \\ -0.1718 & 0.2754 & 0.0190 & -0.0168 & -0.1795 & 0.4047 \\ -0.1498 & -0.2469 & -0.3048 & 0.1890; & & \\ -0.1762 & 0.0352 & -0.0136 & -0.9898 & -1.0683 & 0.0256 \\ 0.1609 & -0.0347 & -0.0367 & 0.0771; & & \\ 0.1899 & -0.1780 & 0.4721 & 0.0818 & -0.0512 & -0.1566 \\ 0.7948 & 0.1826 & 0.1735 & -0.1519; & & \\ 0.0709 & -0.0088 & -0.7242 & 0.0408 & 0.1963 & 0.1012 \\ 0.2193 & 0.0197 & -0.0057 & 1.2080], & & \end{bmatrix}$$

$$\mathbf{b^2} = [-0.0618 \quad -0.6085 \quad 0.6155 \quad 0.0959 \quad 0.6173 \quad 0.0051 \quad -0.0783 \quad 0.4383]',$$

$$\mathbf{LW^{3,2}} = \begin{bmatrix} -0.0497 & -0.9713 & 0.9729 & 0.8669 & 0.9733 & 0.2023 \\ -0.2644 & -0.2404; & & & & \\ -1.2864 & -0.8612 & 0.8711 & 0.3311 & 0.8736 & 0.2509 \\ -0.5259 & 0.2643; & & & & \\ -0.7962 & -0.5790 & 0.5723 & 0.9310 & 0.5705 & 1.7236 \\ -0.5548 & 0.4936; & & & & \\ -0.9444 & -0.7734 & 0.7691 & 0.9663 & 0.7680 & -0.1500 \\ -0.0542 & 1.3300; & & & & \\ -0.1635 & -0.7086 & 0.7167 & 0.2985 & 0.7188 & 0.5962 \\ -0.8770 & 1.4585; & & & & \\ -0.5579 & -0.6606 & 0.6565 & 0.8902 & 0.6554 & -0.2721 \\ -1.2516 & 0.3027], & & & & \end{bmatrix}$$

$$\mathbf{b^3} = [1.1355 \quad 1.1152 \quad 0.4423 \quad 0.7427 \quad 0.9328 \quad 0.5303]'.$$

For instance, if our design attributes from T1 to T10 are listed below:

$$\mathrm{x} = [5 \quad 20 \quad 5 \quad 1900 \quad 8 \quad 8 \quad 33 \quad 900 \quad 2 \quad 15]',$$

we can get their solutions from (1)-(4):

$$\mathbf{a} = [4.9875 \quad 5.8225 \quad 5.3955 \quad 5.2707 \quad 5.2495 \quad 4.8528].$$

Here "**a**" is a vector representing the customer's requirements satisfaction levels which were attained based on the achieved technical targets.

Table 6 shows the technical target setting comparison between the linear regression method and the ANN method. Table 7 shows an evaluation of quality of service requirements in the linear regression method applied to the same design attribute set T as the ANN method. Each element in the T-R matrix in Table 7 is calculated according

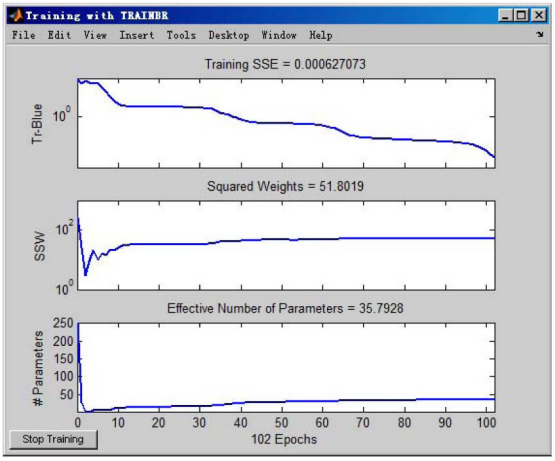



Fig. 10. Training process of ANN for evaluation of satisfaction of quality of service requirements. TRAINBR-calcjx, Epoch 102/1e+006, SSE 0.000627073/0.001, SSW 51.8019, Grad 2.28e-001/1.00e-010, #Par3.58e+001/252 TRAINBR, Performance goal met. $\mathrm{MSE} = 1.7419\text{e-}005$.

TABLE 6
Quality of Service Requirements Evaluation Comparison Between Linear Regression and ANN Methods

| T1: 5<br>T2: 20<br>T3: 5<br>T4 : 1900<br>T5 : 8<br>T6 : 8<br>T7 : 33<br>T8 : 900<br>T9 : 2<br>T10: 15 | R1: InterOperability | R2: Usability | R3: Adaptability | R4: Perfomance | R5: Reliability | R6: Scalability |
|---|---|---|---|---|---|---|
| Linear Regression | 5.1 | 5.9 | 5.5 | 4.7 | 5.0 | 5.0 |
| ANN | 5.0 | 5.8 | 5.4 | 5.3 | 5.2 | 4.9 |
| Difference(%) | -2.0 | -2.0 | -2.7 | 10.8 | 4.9 | -4.0 |

*The quality of service requirement value in linear regression method is calculated as: $R_i = \sum r_{ij} R_{ij}$ , where $r_{ij}$ is design attribute $T_j$'s contribution ratio to $R_i$, as shown in table 3, $R_{ij}$ is the expected value of $R_i$ in terms of design attributes $T_j$, as shown in table 7.*
*The difference = (Linear Regression value –ANN value)/ANN Value*

the relationship listed in Table 4 and multiplied by the corresponding impact ratio listed in Table 3.

Take R4 as an example. The expected level of quality in terms of satisfaction of R4 is calculated as follows, after Rj,4 is obtained using the linear regression method [7], [12]:

$$R_4 = \sum_{j=1}^{10} r_{j,4} R_{j,4},$$

where $r_{j,4}$ is the corresponding impact ratio listed in Table 3, and $R_{j,4}$ is the corresponding evaluation of quality of service requirement listed in Table 7.

In statistics, the mean square error or *MSE* of an estimator is one of many ways to quantify the amount by which an estimator differs from its actual value of the quantity being estimated [35].

TABLE 7
Evaluation of Quality of Service Requirements in the Linear Regression Method

| | T1 | T2 | T3 | T4 | T5 | T6 | T7 | T8 | T9 | T10 |
|---|---|---|---|---|---|---|---|---|---|---|
| | 5 | 20 | 5 | 1900 | 8 | 8 | 33 | 900 | 2 | 15 |
| R1 | 5.03 | | 5.44 | 5.07 | | | | | | |
| R2 | 5.75 | 6.00 | | 5.95 | | | | | | |
| R3 | | | | 5.60 | 5.03 | | | | | |
| R4 | | | 6.60 | | 5.31 | 5.08 | 5.04 | | 4.43 | 5.06 |
| R5 | | | | | | 4.99 | | | | |
| R6 | | | 5.37 | | 4.65 | | | 5.11 | | |

*The data is calculated according to the relationship equation between R and T in Table 4 . The calculated data is more accurate than that, because of the space limitation here.*

$$\begin{aligned} R_4 &= \sum_{j=1}^{10} r_{j,4} R_{j,4} \\ &= r_{3,4} \times R_{3,4} + r_{5,4} \times R_{5,4} + r_{6,4} \times R_{6,4} \\ &\quad + r_{7,4} \times R_{7,4} + r_{9,4} \times R_{9,4} + r_{10,4} \times R_{10,4} \\ &= 0.038 \times 6.60 + 0.038 \times 5.31 + 0.115 \times 5.08 \\ &\quad + 0.346 \times 5.04 + 0.346 \times 4.43 + 0.115 \times 5.06 \approx 4.7. \end{aligned}$$

The *MSE* of R4 in linear regression method can be calculated as follows (each $MSER_{j,4}$ is obtained according the realtionship equation listed in Table 4, and the calucation process is omitted):

$$\begin{aligned} MSE(R_4) &= \sum_{j=1}^{10} r_{j,4} \cdot MSER_{j,4} \\ &= r_{3,4} \times MSER_{3,4} + r_{5,4} \times MSER_{5,4} + r_{6,4} \times MSER_{6,4} \\ &\quad + r_{7,4} \times MSER_{7,4} + r_{9,4} \times MSER_{9,4} + r_{10,4} \times MSER_{10,4} \\ &= 0.038 \times 0.524 + 0.038 \times 0.265 + 0.115 \times 0.019 \\ &\quad + 0.346 \times 0.102 + 0.346 \times 0.040 + 0.115 \times 0.172 = 0.101. \end{aligned}$$

On the other hand, the *MSE(R4)* of the ANN method is less than 1.7419e-005, which is the MSE of all $\mathrm{Ri}(\mathrm{i} = 1, \ldots 6)$ and, as shown in Fig. 10, is much less than the *MSE(R4)* of the impact analysis based linear regression method. Therefore, the ANN method has a much better accuracy.

## 6 Conclusion

QFD enables explicit tracing of customers' needs to design attributes related to web services. Technical targets of a web service system should be set up according to quality goals

and relationships between its quality of service requirements, and web service design attributes when QFD is used to manage its quality and improve its customer satisfaction. Unfortunately, existing technical target setting methods in QFD, such as benchmarking and linear regression method, are based on an assumption that the relationship between satisfaction levels of web service requirements and technical values of design attributes are linear. This assumption sometimes does not hold for many web service systems. In this paper, a new method of technical target setting in QFD is developed for web service systems based on the Bayesian regularized artificial neural network. Technical targets of design attributes identified using the proposed method are consistent with relationships between design attributes and quality of service requirements, no matter whether they are linear or nonlinear. A web application platform is used to illustrate the method and to demonstrate its effectiveness. It enables developers of web service systems in their design phase to determine technical targets of their design attributes based on their customers' requirements.

## REFERENCES


[1] M. Miller, *Cloud Computing: Web-Based Applications That Change the Way You Work and Collaborate Online.* Que, 2008.

[2] E. Newcomer and G. Lomow, *Understanding SOA with Web Services.* Addison-Wesley Professional, 2004.

[3] "Enterprise Software Customer Survey 2007," http://www.software2007.com/grafix/pdf/Enterprise-Software-Customer-Survey-2007.pdf, 2009.

[4] "Enterprise Software Customer Survey 2008," http://www.software2008.com/downloads/mckinsey_software2008_survey.pdf, 2009.

[5] D. Sanderson, *Programming Google App Engine: Rough Cuts Version.* O'Reilly, 2008.

[6] M.H. Ibrahim, K. Holley, N.M. Josuttis, B. Michelson, D. Thomas, and J. deVadoss, "The Future of SOA: What Worked, What Didn't, and Where Is It Going from Here?" *Proc. 22nd ACM SIGPLAN Conf. Object-Oriented Programming Systems and Applications Companion (OOPSLA '07),* pp. 1034-1037, Oct. 2007.

[7] X.F. Liu and L. Zhu, "Design of SOA Based Web Service Systems Using QFD for Satisfaction of Quality of Service Requirements," *Proc. IEEE Int'l Conf. Web Services (ICWS),* pp. 567-574, 2009.

[8] Y. Akao, *Quality Function Deployment.* Productivity Press, 1990.

[9] M. Erder and P. Pureur, "QFD in the Architecture Development Process," *IT Professional,* vol. 5, no. 6, pp. 44-52, 2003.

[10] S. Haag, M.K. Raja, and L.L. Schkade, "Quality Function Deployment—Usage in Software Development," *Comm. ACM,* vol. 39, no. 1, pp. 42-49, 1996.

[11] J.R. Hauser and D. Clausing, "The House of Quality," *Harvard Business Rev.,* vol. 66, no. 3, pp. 63-73, 1988.

[12] X.F. Liu, P. Inuganti, and K. Noguchi, "Technical Target Setting in Time-Stamped Quality Function Deployment," *Total Quality Management and Business Excellence,* vol. 17, no. 2, pp. 149-177, Mar. 2006.

[13] X.F. Liu, K. Noguchi, A. Dhungana, V.V.N.S.N. Srirangam A., and P. Inuganti, "A Quantitative Approach for Setting Technical Targets Based on Impact Analysis in Software Quality Function Deployment (SQFD)," *Software Quality J.,* vol. 14, no. 2, pp. 113-134, June 2006.

[14] X.F. Liu, Y. Sun, G. Kane, Y. Kyoya, and K. Noguchi, "Business-Oriented Software Process Improvement Based on CMM Using QFD," *J. Software Process Improvement and Practice,* vol. 11, no. 6, pp. 573-589, Nov./Dec. 2006.

[15] M. Yang, Y. Li, S. Li, and P. Li., "ANN-Based Fuzzy Reasoning to Determine the Importance of Technical Requirements in QFD," *Proc. Fourth Int'l Conf. Wireless Comm., Networking and Mobile Computing (WiCOM '08),* pp. 1-5, 2008.

[16] F. Siraj et al., "Quality Function Deployment Analysis Based on Neural Network and Statistical Results," *Int'l J. Simulation: Systems, Science and Technology,* vol. 9, no. 2, pp. 73-81, May 2008.

[17] S. Zhang, Y. Dong, B. Pei, and X. Yang, "Research on the Optimization Method of Logistics Service Capacity Based on Dynamic QFD," *Proc. Int'l Conf. Intelligent Computation Technology and Automation,* pp. 664-668, Oct. 2008.

[18] S. Myint, "A Framework of an Intelligent Quality Function Deployment (IQFD) for Discrete Assembly Environment," *Computers and Industrial Eng.,* vol. 45, pp. 269-283, 2003.

[19] Y.-C. Chou, "Applying Neural Networks in Quality Function Deployment Process for Conceptual Design," *J. Chinese Inst. of Industrial Engineers,* vol. 21, no. 6, pp. 587-596, 2004.

[20] D. Paul, S.R. Bhadra Chaudhuri, D. Mukherjee, and S.N. Mandal, "A Soft Computing Model for Optimizing Significant Parameters of Insolation Distribution in BIPV Application," *Proc. Int'l Conf. Computer Science and Information Technology,* Aug. 2008.

[21] K.M. Daws, Z.A. Ahmed, and A.A. Moosa, "An Intelligent Quality Function Deployment (IQFD) for Manufacturing Process Environment," *Jordan J. Mechanical and Industrial Eng.,* vol. 3, no. 1, pp. 23-30, Mar. 2009.

[22] Z.-H. Ren, B.-C. Wang, and B.-C. Wen, "A Model of HoQ Templet Automatic Generation Based on RBF-ANN," *Proc. Third Int'l Conf. Machine Learning Cybernetics,* pp. 3497-3500, 2004.

[23] S. Venkatachalam, C. Arumugam, K. Raja, and V. Selladurai, "Quality Function Deployment in Agile Parallel Machine Scheduling through Neural Network Technique," *Asian J. Scientific Research,* vol. 1, no. 2, pp. 146-152, 2008.

[24] L. O'Brien, L. Bass, and P. Merson, "Quality Attributes and Service-Oriented Architectures," CMU/SEI-2005-TN-014, Sept. 2005.

[25] H. Demuth and M. Beale, *Neural Network Toolbox User's Guide Version 4.* The MathWorks, Inc., 2002.

[26] T. Chen and H. Chen, "Universal Approximation to Nonlinear Operators by Neural Networks with Arbitrary Activation Functions and Its Application to Dynamical Systems," *IEEE Trans. Neural Networks,* vol. 6, no. 4, pp. 911-917, July 1995.

[27] R. Neal, *Bayesian Learning for Neural Networks.* Springer, 1996.

[28] J. Lampinen and A. Vehtari, "Bayesian Approach for Neural Networks—Review and Case Studies," *Neural Networks,* vol. 14, no. 3, pp. 257-274, Apr. 2001.

[29] M. Matteucci and D. Spadoni, "Evolutionary Learning of Rich Neural Networks in the Bayesian Model Selection Framework," *Int'l J. Applied Math. and Computer Science,* vol. 14, no. 3, pp. 423-440, 2004.

[30] F.D. Foresee and M.T. Hagan, "Gauss-Newton Approximation to Bayesian Regularization," *Proc. Int'l Joint Conf. Neural Networks,* pp. 1930-1935, 1997.

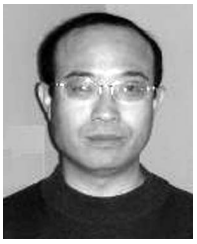
[31] D. Nguyen and B. Widrow, "Improving the Learning Speed of 2-Layer Neural Networks by Choosing Initial Values of the Adaptive Weights," *Proc. Int'l Joint Conf. Neural Networks (IJCNN),* vol. 3, pp. 21-26, July 1990.

[32] P.A.D. Castro and F.J. Von Zuben, "Bayesian Learning of Neural Networks by Means of Artificial Immune Systems," *Proc. Int'l Joint Conf. Neural Networks,* July 2006.

[33] M.S. Khan and P. Coulibaly, "Streamflow Forecasting with Uncertainty Estimate Using Bayesian Learning for ANN," *Proc. Int'l Joint Conf. Neural Networks,* July/Aug. 2005.

[34] K.K. Aggarwal, Y. Singh, P. Chandra, and M. Puri, "Bayesian Regularization in a Neural Network Model to Estimate Lines of Code Using Function Points," *J. Computer Sciences,* vol. 1, no. 4, pp. 505-509, 2005.

[35] http://en.wikipedia.org/wiki/Mean_squared_error, 2010.



**Lianzhang Zhu** received the BS degree in applied mathematics from the China University of Petroleum (East China), Dongying, in 1986, and the MS degree in computer architecture from the Shenyang Computing Institute, Chinese Academy of Sciences in 1989. He received the PhD degree in mineral exploration from the China University of Petroleum (East China), Dongying, in 2003. He is a professor in the College of Computer Science and Communication Engineering, China University of Petroleum. His current research interests are in the areas of computer performance engineering and quality of service of computer systems, and data modeling in oil fields.

**Xiaoqing (Frank) Liu** received the BS degree in computer applications from the Changsha Institute of Technology in 1982, and the MS degree in computer science from Southeast University, Nanjing, China, in 1985. He received the PhD degree in computer science from Texas A&M University, College Station, in 1995. He is a professor in the Computer Science Department at the Missouri University of Science and Technology. He has been working in the areas of software engineering, requirements engineering, software quality management, fuzzy logic, collaborative systems, and database systems since 1982. His research focus is to develop software engineering methods and tools which help software developers to develop high quality software systems on time and within budget. He has published 80 referred technical papers. He is a member of the ACM, the IEEE Computer Society, and the IEEE.