Fuzzy-Neural Controller in Service Requests Distribution Broker for SOA-Based Systems

Mariusz Fras[1], Anna Zatwarnicka[2], and Krzysztof Zatwarnicki[2]

[1] Wroclaw University of Technology, Institute of Informatics,
Wybrzeze Wyspianskiego 27, 50-370 Wroclaw, Poland
mariusz.fras@pwr.wroc.pl
[2] Opole University of Technology, Institute of Automatic Control
and Computer Science, ul. Mikołajczyka 5, 45-271 Opole, Poland
a.zatwarnicka@po.opole.pl, k.zatwarnicki@gmail.com

Abstract. The evolution of software architectures led to the rising importance of the Service Oriented Architecture (SOA) concept. This architecture paradigm support building flexible distributed service systems. In the paper the architecture of service request distribution broker designed for use in SOA-based systems is proposed. The broker is built with idea of fuzzy control. The functional and non-functional request requirements in conjunction with monitoring of execution and communication links are used to distribute requests. Decisions are made with use of fuzzy-neural network.

Keywords: Service Oriented Architecture, service request distribution, fuzzy-neural network.

1 Introduction

One of the most important issues for contemporary computer network service providers is high quality of delivery of network services. Very important subject studied now in many research projects is the evaluation of service quality based on the knowledge on client-to-server network path and service execution system efficiency. Users want to complete requested service without inconvenience and as fast as possible with low latency and high throughput. According to those users activity development of Quality of Web Service (QoWS) sensitive Internet applications raises a need for building Web site services supporting quality and differentiation of services [1]. Nowadays, Service Oriented Architecture (SOA) paradigm gives us more universal approach to composing and building Internet network applications.

The information systems designed with SOA paradigm are working both in local and wide area networks, particularly built with Internet and GRID recourses. The clients don't need to carry about the localization and execution meanings of the service it want to use. The set of available services can change, however this process is not frequent, and usually the set of localizations of given service is constant for some period [2,3].

A. Kwiecień, P. Gaj, and P. Stera (Eds.): CN 2010, CCIS 79, pp. 121–130, 2010.

The paradigm of SOA says that given service can be achievable from different service providers. Because the parameters of provider's servers and communication links to them (communication costs) can be different, the quality of service delivery (especially the time of delivery) can be significantly different too [4]. Taking into account distribution of the services and characteristic of Internet network the proper architecture of service delivery based on service request distribution may significantly increase the quality of services built with SOA paradigm.

In this work the architecture of service request distribution broker built with fuzzy-neural control idea and designed for use in SOA-based systems is proposed.

2 The Request Broker for SOA-Based Systems

The idea of the proposed service request distribution system is to distribute clients requests via service broker B to known execution systems of service providers sp_m (Fig. 1).

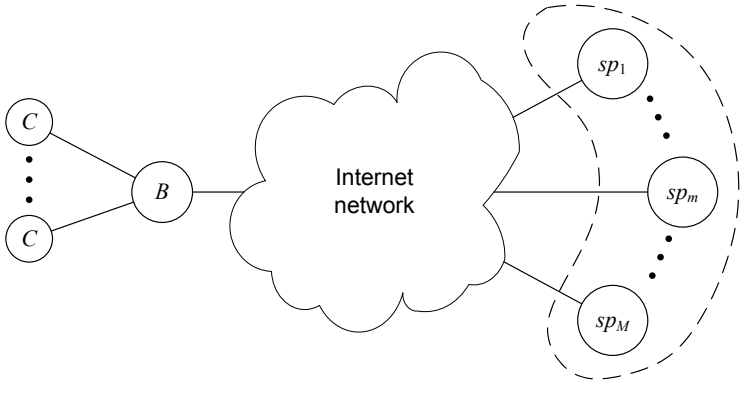

Fig. 1. General infrastructure of service request distribution system

The assumptions for the system are the following:

- the client of the broker is a system delivering complex services $cs(i) \in CS$, where CS is a set of possible services, $i \in \langle 1, \ldots, I \rangle$, aggregated from atomic services $as(j)$, $j \in \langle 1, \ldots, J \rangle$, available in execution systems distributed in the Internet network,
- there is known the set $SP = \{sp_1, \ldots, sp_m, \ldots, sp_M\}$, $m \in \langle 1, \ldots, M \rangle$, of execution systems and it is constant for some considered period,
- there is known the set of atomic services $AS = \{as(1), \ldots, as(j), \ldots, sp(J)\}$ that can be requested by the client and that are available in execution systems $sp_m \in SP$,

- the given atomic service $as(j)$ can be available in several different localizations (execution systems) – the concrete atomic service at given localization is called instance of atomic service,
- the considered resources (from the broker point of view) that affect service execution parameters, are communication links and execution systems.

The goal of the broker is to distribute requests for atomic services that constitute requested complex services (allocate communication resources) in order to fulfill formulated demands for execution of services. Knowing demands on service execution it should distribute request to proper execution system.

Let $CS = \{cs(1), \ldots, cs(i), \ldots, cs(I)\}$, $i \in \langle 1, \ldots, I\rangle$ be the set of possible complex services supplied by the complex service delivery system. The complex service $cs(i)$ is composed with more basic services, especially atomic services $as(k)$, $k \in \langle 1, \ldots, K_i\rangle$, i.e. the ones that can't be partitioned thereafter. The complex service can be described as a directed graph of atomic services $GA\langle V, A\rangle$, where $V = \{as(1), \ldots, as(k), \ldots, cs(K)\}$, and $A \subset V \times V$ is a set of pairs defining edges of execution dependencies between atomic services (vertices) – so called implementation graph of complex service.

The atomic services $as(j), j \in \langle 1, \ldots, J\rangle$ are available (localized) at the constant set SP of execution systems (service providers). The execution system sp_m delivers the set $IS(sp_m) = \{is(m_1), \ldots, is(m_p), \ldots, is(m_P)\}$ of instances of atomic services, where: P – the number of different services.

The instance of the service $is(m_p)$ is characterized by the non-functional parameters $\Psi(is(m_p)) = [\Psi^1_{m_p}, \Psi^2_{m_p}, \ldots, \Psi^f_{m_p}, \ldots, \Psi^F_{m_p}]$, that may be different for two different instances of the same atomic service. An examples of non-functional parameters are: the level of security, the completion time of service execution, etc.

Let $GA_n\langle AS_n, ES_n\rangle$ be the implementation graph of complex service requested at the moment n. The set AS_n is a subset of the set AS containing all atomic services necessary to accomplish complex service cs_n requested at the moment n: $AS_n = \{as(n_1), as(n_2), \ldots, as(n_k), \ldots, as(n_{K_n})\} \subset AS$, $k \in \langle 1, \ldots, K_N\rangle$, where K_n is the number of atomic services (number of vertices in the implementation graph). For each implementation graph it is an equivalent graph $G_n\langle IS_n, ES_n\rangle$, where $IS_n = \{IS_{n1}, IS_{n2}, \ldots, IS_{nk}, \ldots, IS_{nK_n}\} \subset IS$, IS is the set of all instances of atomic services. The set IS_{nk} is the set of all instances of atomic service as_{nk}, i.e. $IS_{nk} = \{is(n,k,1), is(n,k,2), \ldots, is(n,k,m), \ldots, is(n,k,M_{nk})\}$. IS_{nk} corresponds to the set $SP(as(n_k)) = \{sp_m : as(n_k) \in AS(sp_m)\}$, i.e. the set of M_{nk} execution systems that deliver instances of atomic service $as(n_k)$. The graph $G_n\langle IS_n, ES_n\rangle$ defines different execution graphs for the implementation graph $GA_n\langle AS_n, ES_n\rangle$.

Suppose, there is an complex service composed from two subsequent atomic services ($as(1)$, $as(2)$), and $as(1)$ is delivered in two different localizations, $as(2)$ is delivered in three different localizations, i.e. $IS_{n1} = \{is(n,1,1), is(n,1,2)\}$ and $IS_{n2} = \{is(n,2,1), is(n,2,2), is(n,2,3)\}$. The complex service can be executed in six different ways what illustrates paths of execution in Fig. 2 (the index n is omitted).

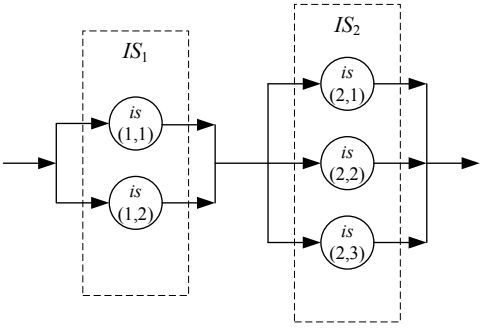

Fig. 2. Possible paths of execution of an example complex service

The choice of specific execution path is a problem that is involved with specifying demands $SLA_{nf,n}$ on non-functional parameters $\Psi(cs_n)$ of the complex service cs_n, that can be specified in request $X_n = \langle GA_n\langle AS_n, ES_n\rangle, SLA_{nf,n}\rangle$ at the moment n. The non-functional parameters of complex service $\Psi(cs_n)$ correspond to non-functional parameters of instances of atomic services $\{\Psi(is(n,k,m)\}$. Knowing $\{\Psi(is(n,k,m)\}$ it is possible to perform different procedures of fulfilling $SLA_{nf,n}$. The general goal of service request distribution system is to determine execution graph of complex service that fulfills $SLA_{nf,n}$:

$$GA_n\langle AS_n, ES_n\rangle \rightarrow G_n^*\langle IS_n^*, ES_n\rangle \ ,$$

where: $IS_n^* \leftarrow min_{k=1,\ldots,K_n} h(\Psi(is(n,k,m))$, what denotes the problem of finding appropriate instances of atomic services that should be requested in order to fulfill formulated demands on non-functional parameters, $h(\Psi(is(n,k,m))$ is formulated criterion of fulfilling $SLA_{nf,n}$.

One of the most important non-functional parameters is completion time of delivery of requested service. The impact on this time have two components communication links and execution systems, specifically completion time of data transfer and completion time of service execution at the execution system. Thus, one of the main functionality of service request distribution broker must to be estimation (forecasting) of non-functional parameters $h(\Psi(is(n,k,m))$ at the moment n for all instances of atomic service, $j = 1,\ldots,J$, $m = 1,\ldots,M$. Especially estimation of service execution time at execution systems and service request/reply transfer time.

To determine for the request arriving at the moment n destination server which should serve this request on the basis on time criteria, the broker has to support the following functionality:

- monitoring of servicing client requests,
- prediction of selected communication link parameters,
- estimation of transfer times and execution times of all instances of atomic services, on the basis of adaptive models of communication links and execution systems.

In the Figure 3 we propose the general architecture of the broker that fulfills all these functionalities.

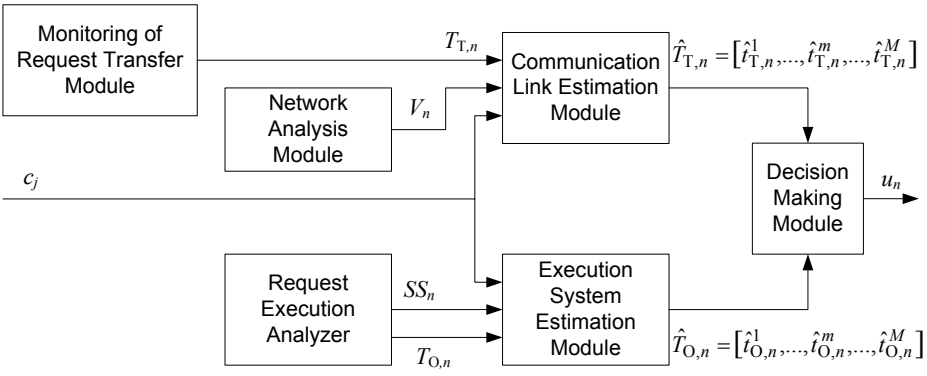

Fig. 3. General architecture of the service request broker

The decision u_n of the choice of one of M execution systems is performed on the basis of the vector $\hat{T}_{\mathrm{T},n}$ of estimated transfer times $\hat{t}^m_{\mathrm{T},n}$ of the request to each server m, and vector $\hat{T}_{\mathrm{O},n}$ of estimated execution times $\hat{t}^m_{\mathrm{O},n}$ of the request at each execution system m. The key components of the broker are estimation modules of the communication links and execution systems. Modules are built of M models of links or execution systems respectively that works as controllers.

The next section describes proposed method for estimation of completion times of data transfers t_{T} for requested atomic services.

3 The Fuzzy-Neural Controller

The input of the communication link model is identifier c_j of requested atomic service that arrives at the moment n, measured real transfer times $t^m_{\mathrm{T},n}$ of prior (before the moment n) requests to the system m and the vector V_n of two communication link parameters: forecasted link throughput $\hat{th}_n$ and forecasted link latency $\hat{t}_{\mathrm{TCPC},n}$ (namely TCP Connect Time). These two parameters are derived from periodic measurements of latency and link throughput, and using time series analysis based prediction algorithms. The output of each model is estimated transfer time $\hat{t}^m_{\mathrm{T},n}$ of the request.

Each communication link model is built for each destination and for each atomic service (for each instance of atomic service). It is designed as fuzzy-neural controller (Fig. 4) based on [5,6,7]. Hereafter, the indexes of link and service are omitted.

The inputs a_n and b_n of the fuzzyfication block are forecasted link latency $\hat{t}_{\mathrm{TCPC},n}$ and forecasted link throughput $\hat{th}_n$ at the moment n. The outputs are the grades of membership of latency μ_A in defined fuzzy sets $Z^a_k, k = 0, \ldots, K$ for

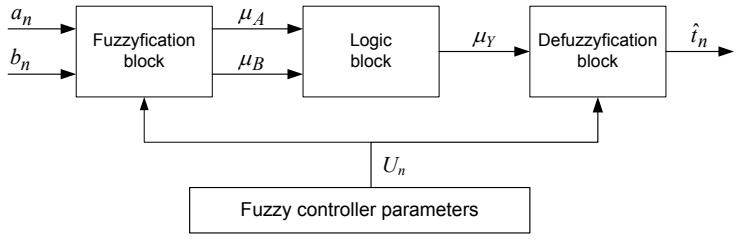

Fig. 4. Communication link model as a fuzzy-neural controller

latency, and grades of membership of throughput μ_B in defined fuzzy sets $Z_l^b, l = 0, \ldots, L$ for throughput. The sets are described with membership functions as follows:

$$\mu_{Z_1^d}(d) = \begin{cases} \frac{\delta_0 - d}{\delta_0} & \text{for } 0 \le d < \delta_0 \\ 0 & \text{for } \delta_0 \le d \end{cases}$$

$$\mu_{Z_h^d}(d) = \begin{cases} \frac{d-\delta_{h-1}}{\delta_h - \delta_{h-1}} & \text{for } \delta_{h-1} < d \le \delta_h \\ \frac{d-\delta_{h+1}}{\delta_h - \delta_{h+1}} & \text{for } \delta_h < d < \delta_{h+1} \\ 0 & \text{for } \delta_{h+1} \le d \end{cases} \tag{1}$$

$$\mu_{Z_H^d}(d) = \begin{cases} 0 & \text{for } 0 \le d \le \delta_{H-1} \\ \frac{d-\delta_{H-1}}{\delta_H - \delta_{H-1}} & \text{for } \delta_{H-1} < d < \delta_H \\ 1 & \text{for } \delta_H \le d \end{cases} .$$

Variable d corresponds to input a or b. Parameters $\delta_0, \ldots, \delta_h, \ldots, \delta_H$ correspond to parameters of fuzzy sets of input a: $\alpha_0, \ldots, \alpha_k, \ldots, \alpha_K$, or parameters of fuzzy sets of input b: $\beta_0, \ldots, \beta_l, \ldots, \beta_L$ respectively (H is equal K or L, h is the index of fuzzy set (k or l). Functions $\mu_{Z_h^d}(d), h = 1, \ldots, H$, are membership functions of fuzzy sets for parameter a i.e. $\mu_{Z_k^a}(a), k = 1, \ldots, K$, or parameter b i.e. $\mu_{Z_l^b}(b), l = 1, \ldots, L$.

The logic block inferences output. It computes grades of membership of consequence μ_Y of the rules that are of the format proposed in [8]:

$$a = Z_k^a \text{ AND } b = Z_l^b \text{ THEN } y = Z_r^y \ , \tag{2}$$

where: Z_r^y is a fuzzy set for output y (linguistic variable for estimated $\hat{t}_n$) $r = 1, \ldots, R, R = K * L$. The membership function for Z_r^y is described by formula:

$$\mu_{Z_r^y}(y) = \begin{cases} 1 \text{ for } y = y_r \\ 0 \text{ for } y \ne y_r \end{cases} . \tag{3}$$

Assuming Mamdani model of inference and Larsen form of T-norm for fuzzy implication, the membership function for consequence of fuzzy rule r is described by:

$$\mu_{Y_r}(y) = \begin{cases} \mu_{R_r}(a^*, b^*) & \text{for } y = y_r \\ 0 & \text{for } y \ne y_r \end{cases} , \tag{4}$$

where $\mu_{R_r}(a^*, b^*) = \mu_{Z_k^a}(a^*) * \mu_{Z_l^b}(b^*)$, a^* and b^* are spice values of inputs. The rule is fired when $\mu_{R_r}(a^*, b^*) > 0$.

After the defuzzyfication (using height method) the estimated transfer time of the request is derived according to formula (5):

$$\hat{t}_n = \sum_{r=1}^{R} \mu_{R_r}(a^*, b^*) * y_r \ . \tag{5}$$

In order to satisfy the need of working of the controller in adaptive mode it is built as a 3-layered neural network similar to [8,9,6] (Fig. 5). The first layer of the network constitutes the fuzzyfication block. Each neuron transform sharp value of the input into fuzzy value. The second layer computes grades of membership for consequences of fuzzy rules. In the third layer and aggregation neuron the sharp value of the input is derived. A, B and Y are vectors of parameters α_k, β_l and y_r.

The parameters α_k and β_l of fuzzy sets for inputs a and b, as well as y_r for output are tuned during learning process using back propagation method [10] on the basis on measured actual request transfer times t_n. The computation formula for output is following (g is the moment of adjustment of parameters after completion of request x_n):

$$y_{r(g+1)} = y_{r(g)} + \gamma_y(\hat{t}_n - \hat{t}_n) * \mu_{R_r(g)}(a^*, b^*) \ , \tag{6}$$

where γ_y is learning constant – scaling factor for parameter y used in learning process.

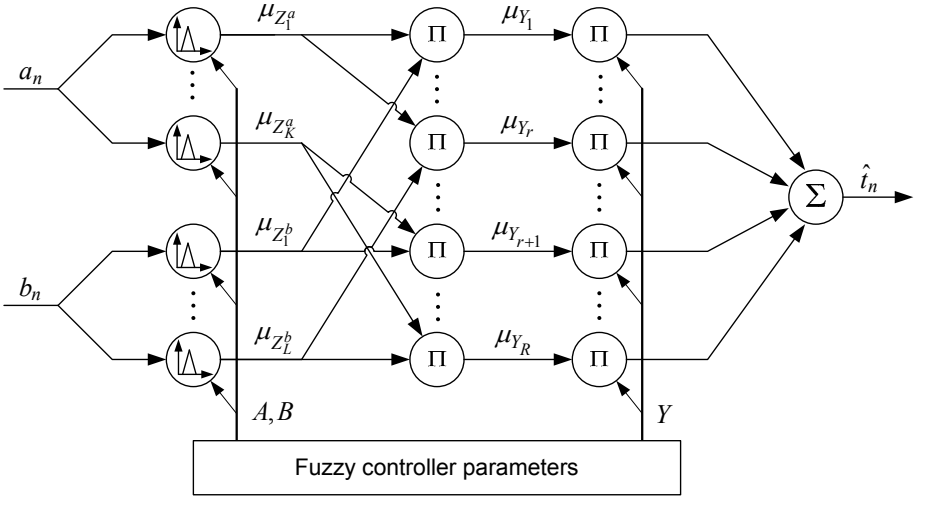

Fig. 5. Fuzzy-neural controller as fuzzy-neural network

The parameters for inputs a and b are tuned according to formulas (7):

$$\begin{aligned} \alpha_{k(g+1)} &= \alpha_{k(g)} + \Delta^{\alpha}_{k(g)} \ , \\ \beta_{k(g+1)} &= \beta_{k(g)} + \Delta^{\beta}_{k(g)} \ , \end{aligned} \tag{7}$$

where:

$$\Delta^{\alpha}_{k(g)} = \gamma_a \cdot (\hat{t_n} - t_n) \cdot \sum_{r=1}^{R} \left[y_{r,n} \cdot \left(\mu_{Z^b_l(g)}(a^*) \right)^{(r)} \right] \cdot \sum_{l=1}^{L} \left(\frac{\partial \mu_{Z^a_k(g)}}{\partial \alpha_{k(g)}} \right) \ , \tag{8}$$

and:

$$\Delta^{\beta}_{k(g)} = \gamma_b \cdot (\hat{t_n} - t_n) \cdot \sum_{r=1}^{R} \left[y_{r,n} \cdot \left(\mu_{Z^a_k(g)}(a^*) \right)^{(r)} \right] \cdot \sum_{l=1}^{L} \left(\frac{\partial \mu_{Z^b_l(g)}}{\partial \beta_{l(g)}} \right) \ , \tag{9}$$

where $\left(\mu_{Z(.)(g)}(.)\right)^{(r)}$ is the value of grade of membership for input for interconnection r (see Fig. 5). Parameters a and b are learning constants.

4 Experimental Measurements for Request Transfer Time Estimation

The components of the request broker that concern estimation of transfer time including fuzzy-neural controller, were implemented as complex set of tools in Linux system. The network analysis module performs active measurements in Internet network and forecast TPC Connect time (latency of communication) and TCP/HTTP throughput on the basis of time series analysis method – Exponentially Weighted Moving Average (EWMA) algorithm. The monitoring module is able to measure transfer times of clients HTTP requests. The fuzzy-neural controller works for several values of scaling factors (learning constant used in learning process for parameters of fuzzy sets, for inputs: throughput and latency) simultaneously.

The preliminary experiments were performed as follows:

- at the client side (localized at Wroclaw University of Technology) the HTTP requests to given set of HTTP servers distributed in Internet network were generated,
- according to preliminary nature of the experiment there were selected 8 servers, for which communication links were the most stable,
- the service requests for transfer static HTTP object (2 MB size) were generated every 5 min. during about 30 hours,
- there were measured HTTP throughput and TCP Connect time for every request,
- the prediction of above parameters were performed using EWMA method,
- the fuzzy-neural controller has estimated transfer times with use of scaling factor $\gamma_a = \gamma_b = \gamma$ equal 0.1, 0.2,. . . , 0.9 simultaneously.

In the Figure 6 the aggregated Mean Average Percentage Error (MAPE) for 150 last measurements is presented. The graph shows data for learning constant γ used in fuzzy-neural controller equal 0.1,..., 0.9. This data shows effectiveness of the estimation when controller works after first phase of learning process. The results suggest that during stable phase of work more optimal parameter of the controller is $\gamma = 0.5$.

In the Figure 7 the cumulated MAPE of request transfer time prediction versus number of request n is shown for learning constant γ equal 0.5. The results shows proper learning of the system. However the error doesn't decline as fast and regularly as we would like. The more detailed analysis of measurements results suggests, that more accurate prediction method of link throughput should improve the work of the controller.

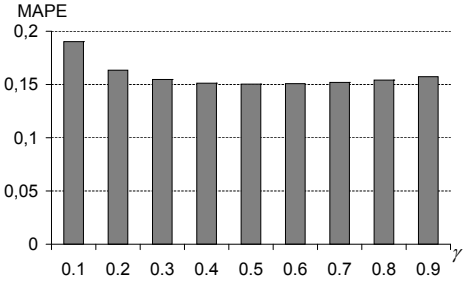

Fig. 6. Cumulative MAPE versus number of request n for learning constant $\gamma = 0.5$

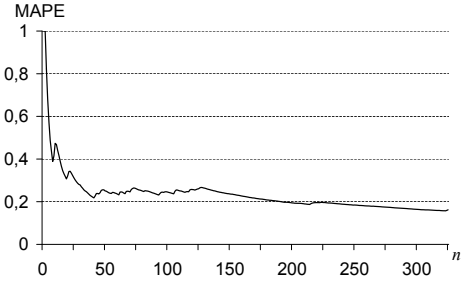

Fig. 7. Aggregated MAPE for 150 last measurements, for learning constant $\gamma = 0.1, \ldots, 0.9$

5 Conclusion

The new approach to build systems in compliance with SOA paradigm raises new challenges in delivery of Internet network services. In this work the architecture of broker designed for use in SOA-based systems has been proposed. The broker support efficient aggregation of complex services composed with atomic ones

that are localized in execution systems distributed in the Internet network. The broker performs requests distribution using fuzzy-neural network thus trying to deliver needed service to the client in the fastest way. The estimation of service transfer time is done with use of designed fuzzy-neural controller.

Presented distribution system was implemented in Linux system. The preliminary experiments based on the real measurements in Internet network show that such approach could help delivering of services in SOA-based systems. However, it should be done by appropriate prediction (or measurement) of communication link parameters. Further research on fuzzy-neural network parameters should also be continued.

Acknowledgements. The research presented in this paper has been partially supported by the European Union within the European Regional Development Fund program no. POIG.01.03.01-00-008/08.

References

1. Cardellini, V., Casalicchio, E., Colajanni, M., Mambelli, M.: Web switch support for Differentiated Services. ACM Perv. Eval. Rev. 29(2), 14–19 (2009)
2. Brawn, P.C.: Implementing SOA. Pearson Education, London (2008)
3. Mabrouk, M.: SOA fundamentals in a nutshell. IBM Corp. (2008)
4. Williams, W., Herman, R., Lopez, L.A., Ebbers, M.: Implementing CICS Web Services. IBM Readbook (2007)
5. Driankov, D., Hellendoorn, H., Reinfrank, M.: An Introduction to Fuzzy Control. Springer, Berlin (1993)
6. Mamdani, E.H.: Application of Fuzzy Logic to Approximate Reasoning Using Linguistic Synthesis. IEEE Trans. on Comp. C-26 (1997)
7. Yager, R.R., Filev, D.P.: Essentials of Fuzzy Modeling and Control, New York (1994)
8. Borzemski, L., Zatwarnicka, A., Zatwarnicki, K.: Global distribution of HTTP requests using the fuzzy-neural decision-making mechanism. In: Proc. of 1st Int. Conf. on Comp. Collective Intelligence. Lect. Notes in AI. Springer, Heidelberg (2009) (in press)
9. Jain, L.C., Martin, N.M.: Fusion of neural networks, fuzzy sets and genetic algorithms industrial applications. CRC Press LLC, London (1999)
10. Keong-Myung, L., Dong-Hoon, K., Hyung, L.: Tuning of fuzzy models by fuzzy neural networks. Fuzzy Sets and System 76 (1995)