HET-NETs 2010
ISBN XXX–XXX
pp. xx–xx

QoS-aware infrastructure resources allocation in systems based on service-oriented architecture paradigm

ADAM GRZECH [a] PIOTR RYGIELSKI [a] PAWEŁ ŚWIĄTEK [a]

[a]Institute of Computer Science
Wroclaw University of Technology, Poland
{adam.grzech, piotr.rygielski, pawel.swiatek}@pwr.wroc.pl

Abstract: In this paper the task of communication and computational resources allocation in systems based on SOA paradigm is considered. The task of resources allocation consists in assigning resources to each incoming service request in such a way, that required level of the quality of service is met. Complex services performance time in distributed environment is assumed as the quality of service measure. Models of the system and analysis of service response time presented in this paper allowed to formulate several tasks of resource allocation in terms of well known quality of service assurance models: *best effort*, *IntServ* and *DiffServ*. For each formulated task solution algorithms were proposed and their correctness was evaluated by means of simulation.

Keywords: quality of service (QoS), service-oriented architecture (SOA), resource allocation, response time guaranties

1. Introduction

Resource allocation and quality of service management in systems based on service-oriented architecture (SOA) paradigm are very important tasks, which allow to maximize satisfaction of clients and profits of service provider [1]. In nowadays SOA systems, which utilize Internet as the communication bus the problem of service response time guaranties arises. Since overall service response time consists of communication and computational delays the task of delivering requested service response time requires proper management of both communication and computational resources [4].

In this work three methods for satisfying quality of service requirements based on well known quality of service assurance models (i.e.: *best effort*, Integrated Services (*IntServ*) and Differentiated Services (*DiffServ*)) [8] are presented.

Paper is organized as follows. In section 2 models of system, complex service and network traffic generated service requests are presented. Section 3 covers qualitative analysis of service response time in the considered system. Basing on assumed models and performed analysis three tasks and solution algorithms for resource allocation for the purpose of quality of service assurance are presented in section 4. Exemplary results of performed simulations are presented in section 5. Finally conclusions are drawn and directions for future research are given in section 6.

2. Model of the system

It is assumed that the considered system delivers complex services composed of atomic services; the latter is defined as a service with an indivisible functionality offered by known and well-specified place or places in the system. Moreover, it is also assumed that each atomic service is available in several versions; different versions of the particular atomic service offer exactly the same functionality and are differentiated by values of various attributes assign to each atomic service [4].

2.1. Complex service composition

Let us assume that AS is the set of all atomic services and contains m separate subsets AS_i ($i = 1, \ldots, m$); $AS = \{AS_1, \ldots, AS_m\}$. Subset AS_i contains all already existing and ordered versions of the particular i-th atomic service as_{ij}; $AS_i = \{as_{i1}, \ldots, as_{in_i}\}$ where n_i is the number of all distinguishable versions of the i-th atomic service. Different versions of all i-th atomic services as_{ij_i} ($j_i = 1, \ldots, n_i$) may be labeled using values of many different attributes (e.g. location within the system, attached security mechanisms, frequency of use, computational complexity, completing time, quality of interface, etc.).

Let us also assume that atomic services from set AS are used to complete complex services s_k ($s_k \in S$, $k = 1, \ldots, K$) in such a way, that each k-th complex service (k-th path through the serially ordered set of atomic services sets) is composed of exactly m different atomic services as_{ij_i} executed one-by-one following increasing values of indexes i ($i = 1, \ldots, m$). Each complex service path s_k is precisely defined by a sequence of indices j_i of particular versions of atomic services used by complex service s_k:

$$s_k = (j_1, \ldots, j_m). \tag{1}$$

The set S of all possible complex services is defined as:

$$S = \{(j_1, \ldots, j_m) : j_1 \in \{1, \ldots, n_1\}, \ldots, j_m \in \{1, \ldots, n_m\}\}. \tag{2}$$

Such defined set of atomic services as well as assumed way of complex service composition means that the available set of m atomic services in various versions allows to obtain K different complex services where $K = n_1 \times \ldots \times n_m$ (see fig. 1).

2.2. Network traffic

In the SOA paradigm it is assumed, that atomic services providing certain functionalities are performed in a distributed environment [6]. Let req_l ($l = 1, 2, \ldots$) represents certain l-th complex service request incoming to the system. In order to deliver requested complex

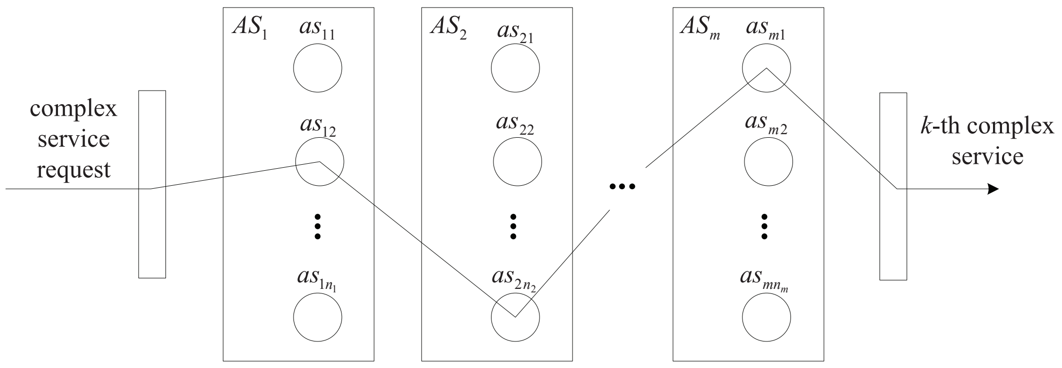


Fig. 1. Complex service composed of serially executed atomic services.

functionality service request req_l has to pass through all m atomic services along certain service path s_k. Obviously atomic services belonging to chosen service path may be placed in remote locations. Choosing certain service path s_k for service request req_l means that request req_l needs to travel from service requester SR through each link and atomic service on service path s_k and back to service requester SR (see fig. 2).

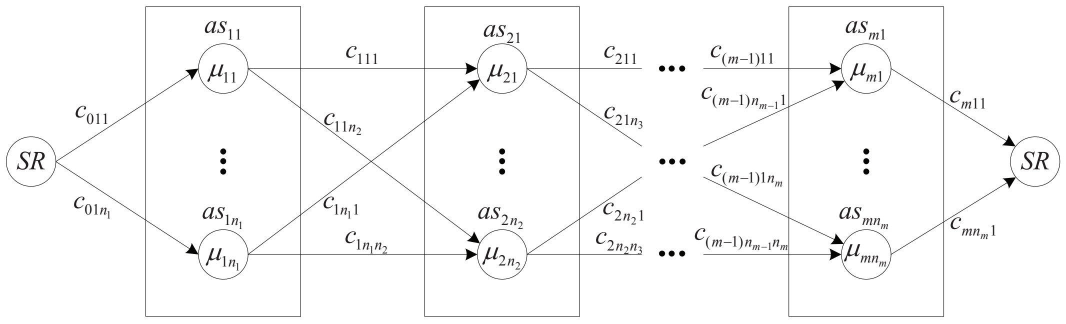


Fig. 2. Atomic services in distributed environment.

Denote by $c_{(i-1)j_{i-1}j_i}$ ($i = 1, \ldots, m, j_{i-1} = 1, \ldots, n_{i-1}, j_i = 1, \ldots, n_i$) the capacity of a communication link connecting two consecutive atomic services $as_{(i-1)j_{i-1}}$ and as_{ij_i}. Parameters c_{01j_1} ($j_1 = 1, \ldots, n_1$) and $c_{mj_m 1}$ ($j_m = 1, \ldots, n_m$) denote respectively: capacities of links between service requester SR and atomic services in the first stage and atomic services in last (m-th) stage and service requester SR.

Execution of each atomic service changes the size of service request in proportional manner:

$$u_{out} = \alpha_{ki} \cdot u_{in}, \tag{3}$$

where u_{in} and u_{out} denote input and output size of service request and proportional coefficient α_{ki} depends on the service path s_k ($k = 1, \ldots, K$) of service request and service request processing stage AS_i ($i = 1, \dots, m$). Input and output size of request can be interpreted as the amount of input and output data necessary to execute and being a result of execution of an atomic service on certain service path. Therefore, each service path s_k is described by vector $\boldsymbol{\alpha}_k = [\alpha_{k1} \dots \alpha_{km}]^T$ of proportional coefficients concerning size of data generated by atomic services along this path.

Let p_k ($k = 1, \dots, K$) denote the probability, that certain service request req_l is served along k-th service path s_k. Moreover, denote by $\overline{u}$ average size of incoming requests, by λ average request arrival rate, and by μ_{ij} average service rate of each atomic service as_{ij}.

Average of traffic $f_{(i-1)j_{i-1}j_i}$ flowing between two consecutive atomic services $as_{(i-1)j_{i-1}}$ and as_{ij} is a sum of traffic generated on service paths passing through atomic services $as_{(i-1)j_{i-1}}$ and as_{ij}:

$$f_{(i-1)j_{i-1}j_i} = \lambda \overline{u} \sum_{k \in K_{(i-1)j_{i-1}j_i}} p_k \prod_{n=1}^{i-1} \alpha_{nk}, \tag{4}$$

where $K_{(i-1)j_{i-1}j_i}$ is a set of indices of service paths passing through atomic services $as_{(i-1)j_{i-1}}$ and as_{ij} and is defined as:

$$K_{(i-1)j_{i-1}j_i} = \{k \in K : as_{(i-1)j_{i-1}}, as_{ij_i} \in s_k\}. \tag{5}$$

Average traffic incoming to certain atomic service as_{ij_i} is a sum of traffic on links incoming to as_{ij_i}:

$$f_{ij_i} = \sum_{j_{i-1}=1}^{n_{i-1}} f_{(i-1)j_{i-1}j_i} = \lambda \overline{u} \sum_{j_{i-1}=1}^{n_{i-1}} \sum_{k \in K_{(i-1)j_{i-1}j_i}} p_k \prod_{n=1}^{i-1} \alpha_{nk}, \tag{6}$$

Average size of traffic f_{k^*} flowing through each k^*-th service path can be calculated as a sum of traffic sizes flowing between consecutive atomic services as_{ij_i} ($i = 1, \dots, m$) along k^*-th path:

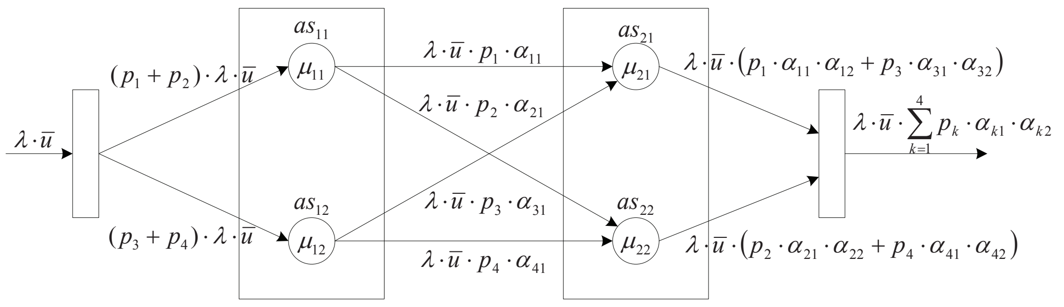
$$f_{k^*} = \sum_{i=1}^{m+1} f_{(i-1)j_{i-1}j_i} = \lambda \overline{u} \sum_{i=1}^{m+1} \sum_{k \in K_{(i-1)s_{k^*}(i-1)s_{k^*}(i)}} p_k \left(1 + \prod_{n=1}^{i-1} \alpha_{nk}\right), \tag{7}$$

where $s_{k^*}(i)$ denotes index j_i of atomic service on i-th stage along path s_{k^*}.

An exemplary system, which consists of two atomic services ($m = 2$), each having two versions ($n_1 = n_2 = 2$) is presented on figure 3. All four possible service paths are enumerated as follows: $s_1 = (as_{11}, as_{21})$, $s_2 = (as_{11}, as_{22})$, $s_1 = (as_{12}, as_{21})$, $s_1 = (as_{12}, as_{22})$. Amount of traffic flowing through each link and each service path is presented on figure 3.

3. Service request response time

Each of atomic services as_{ij_i} ($i = 1, \dots, m, j_i = 1, \dots, n_i$) and communication links $c_{(i-1)j_{i-1}j_i}$ ($i = 1, \dots, m, j_{i-1} = 1, \dots, n_{i-1}, j_i = 1, \dots, n_i$) between atomic services



as_{11}
as_{12}
as_{21}
as_{22}
μ_{11}
μ_{12}
μ_{21}
μ_{22}
$\lambda \cdot \overline{u}$
$(p_1 + p_2) \cdot \lambda \cdot \overline{u}$
$(p_3 + p_4) \cdot \lambda \cdot \overline{u}$
$\lambda \cdot \overline{u} \cdot p_1 \cdot \alpha_{11}$
$\lambda \cdot \overline{u} \cdot p_2 \cdot \alpha_{21}$
$\lambda \cdot \overline{u} \cdot p_3 \cdot \alpha_{31}$
$\lambda \cdot \overline{u} \cdot p_4 \cdot \alpha_{41}$
$\lambda \cdot \overline{u} \cdot (p_1 \cdot \alpha_{11} \cdot \alpha_{12} + p_3 \cdot \alpha_{31} \cdot \alpha_{32})$
$\lambda \cdot \overline{u} \cdot (p_2 \cdot \alpha_{21} \cdot \alpha_{22} + p_4 \cdot \alpha_{41} \cdot \alpha_{42})$
$\lambda \cdot \overline{u} \cdot \sum_{k=1}^{4} p_k \cdot \alpha_{k1} \cdot \alpha_{k2}$

Fig. 3. Amount traffic flowing through considered system.

$as_{(i-1)j_{i-1}}$ and as_{ij_i} can be treated as single queue single processor queuing systems with average service rate μ_{ij_i} and $c_{(i-1)j_{i-1}j_i}$ respectively. Assuming, that incoming stream of requests is a Poisson stream and that atomic services and links service rates are characterized by exponential distributions with respective parameters μ_{ij_i} and $c_{(i-1)j_{i-1}j_i}$, average response times of atomic services and links are given by [3]:

$$\overline{d}_{ij_i} = (\mu_{ij_i} - f_{ij_i})^{-1} \tag{8}$$

and

$$\widehat{d}_{(i-1)j_{i-1}j_i} = (c_{(i-1)j_{i-1}j_i} - f_{(i-1)j_{i-1}j_i})^{-1} \tag{9}$$

respectively, where f_{ij_i} is the average intensity of traffic incoming to atomic service as_{ij_i} (eq. 6) and $f_{(i-1)j_{i-1}j_i}$ is the average traffic intensity on the link connecting atomic services $as_{(i-1)j_{i-1}}$ and as_{ij_i} (eq. 4). It can be shown (e.g. [2, 3]) that under assumption of Poisson request arrivals and exponential service rates probability distributions of atomic services and links response times are exponential with parameters $\overline{d}_{ij_i}$ and $\widehat{d}_{(i-1)j_{i-1}j_i}$ respectively.

Denote by r_k random variable representing response time of certain complex service request req_l serviced along k-th service path s_k. Random variable r_k is a sum of random variables representing response times of atomic services as_{ij_i} ($\overline{r}_{ij_i}$) and links $c_{(i-1)j_{i-1}j_i}$ ($\widehat{r}_{(i-1)j_{i-1}j_i}$) belonging to service path s_k:

$$r_k = \sum_{i=1}^{m} \overline{r}_{ij_i} + \sum_{i=1}^{m+1} \widehat{r}_{(i-1)j_{i-1}j_i}. \tag{10}$$

Denote by $\boldsymbol{\delta}_k = [\delta_{k1} \ldots \delta_{k2m+1}]$ vector of parameters $\overline{d}_{ij_i}$ and $\widehat{d}_{(i-1)j_{i-1}j_i}$ of random variables $\overline{r}_{ij_i}$ and $\widehat{r}_{(i-1)j_{i-1}j_i}$ such that $\delta_{k1} = \overline{d}_{1j_1}, \ldots, \delta_{km} = \overline{d}_{mj_m}, \delta_{km+1} = \widehat{d}_{(0)j_0j_1}, \ldots, \delta_{k2m+1} = \widehat{d}_{(m)j_mj_{m+1}}$. Since $\overline{r}_{ij_i}$ and $\widehat{r}_{(i-1)j_{i-1}j_i}$ are exponentially distributed with different parameters, probability distribution of r_k is given as [2]:

$$f_{rk}(r_k) = \left[\prod_{i=1}^{2m+1} \delta_{ki}\right] \sum_{i=1}^{2m+1} \frac{e^{-\delta_{ki} r_k}}{\prod_{j \neq i} (\delta_{kj} - \delta_{ki})}, r_k > 0 \tag{11}$$

Cumulative distribution function of complex service response time is given by integral:

$$F_{rk}(r_k) = \int_0^{r_k} f_{rk}(x)dx = \left[\prod_{i=1}^{2m+1} \delta_{ki}\right] \sum_{i=1}^{2m+1} \frac{1 - e^{-\delta_{ki} r_k}}{\delta_{ki} \prod_{j \neq i} (\delta_{kj} - \delta_{ki})}, r_k > 0 \qquad (12)$$

Functions $f_{rk}(r_k)$ and $F_{rk}(r_k)$ denote respectively probability and cumulative distribution functions of complex service response time r_k for requests served along k-th service path s_k.

4. Task of resource allocation

Models presented in section 2. and service response time analysis presented in section 3. allows to formulate various resource allocation tasks, the aim of which is to deliver required level of the quality of services (QoS) measured as complex service response time. In this paper we focus on three basic tasks: task of minimization of the average service response time, task of delivering required service response time, and task of delivering average service response time for different classes of service requests. Presented tasks of resource allocation can be better understood in terms of quality of service assurance models known from computer communication network theory [7], namely: best effort, IntServ and DiffServ models.

4.1. Average service response time minimization (best effort)

Average response time $\overline{d}_k$ experienced by service requests on k-th service path s_k is the sum of average delays experienced on each link and atomic service belonging to service path s_k:

$$\overline{d}_k = \sum_{i=1}^{m} \overline{d}_{ij_i} + \sum_{i=1}^{m+1} \widehat{d}_{(i-1)j_{i-1}j_i}, \qquad (13)$$

where $j_i \in s_k$ for $i = 0, \ldots, m$. Under assumption of Poisson request arrivals and exponential link and atomic service response times average response time $\overline{d}_k$ can be calculated more precisely as the expected value of complex service response time $E[r_k]$:

$$\overline{d}_k = E[r_k] = \int_{r_k=0}^{\infty} r_k f_{rk}(r_k) dr_k, \qquad (14)$$

where $f_{rk}(r_k)$ is defined by equation (11).

Average response time $\overline{d}$ experienced by service requests in whole system can be calculated as the weighted average over response times of each path s_k ($k = 1, \ldots, K$):

$$\overline{d} = \sum_{k=1}^{K} p_k \overline{d}_k, \qquad (15)$$

where p_k is the probability, that certain service request will be served along k-th service path s_k.

The aim of the task of minimization of the average service response time is to find such a vector $\mathbf{p} = [p_1 \ldots p_K]$ of probabilities of choosing different service paths s_k for which average service response time is minimized:

$$\mathbf{p}^* = \arg\min_{\mathbf{p}} \sum_{k=1}^{K} p_k \overline{d}_k, \tag{16}$$

with respect to constraints on probabilities $\mathbf{p}$:

$$\sum_{k=1}^{K} p_k = 1 \quad \text{and} \quad p_k \geq 0 \ \text{ for } \ k = 1, \ldots, K. \tag{17}$$

Since average response time $\overline{d}_k$ of each service path s_k depend on request arrival intensity λ, average request size $\overline{u}$ and probabilities $\mathbf{p}$, which change over time, optimization task (16) has to be solved iteratively in consecutive time steps. Resource allocation consists in assigning incoming request to service paths in such a way, that intensities of requests served along each service path are proportional to calculated probabilities $\mathbf{p}^*$. For large number K of service paths this approach may be inefficient due to high computational complexity of optimization task (16). In such a case one can approximate optimal allocation by application of greedy approach, which for each new service request chooses service path s_{k^*} with the lowest average delay $\overline{d}_{k^*}$:

$$k^* = \arg\min_{k} \overline{d}_k. \tag{18}$$

4.2. Service response time guaranties (IntServ)

Denote by $S(t_l)$ state of the system at the moment t_l of arrival of new request req_l. State $S(t_l)$ contains information concerning moments of arrival, assigned service paths and location of all service request present in the system at moment t_l. Given system state $S(t_l)$ it is possible to calculate exact service response time $d_k(req_l)$ for request req_l for each service path s_k:

$$d_k(req_l) = d(S(t_l), req_l, k), \tag{19}$$

where function $d(S(t_l), req_l, k)$ (presented in [4]) represents an iterative algorithm for calculation of response time of service request req_l along k-th service path.

In the task of delivering quality of service it is assumed that each incoming request req_l contains a vector $\mathbf{q}_l$ of requirements concerning values of various parameters describing quality of service. Besides required service response time d_l^* vector $\mathbf{q}_l$ may contain parameters describing: security, cost, availability, etc.

The aim of the task of guarantying service response time is to find such a service path s_k for which service response time requirements are satisfied:

$$k^* = \max_{k} \left\{ k \in \{1, \ldots, K\} : d_k(req_l) \leq d_l^* \right\}. \tag{20}$$

It is possible that there does not exist such a path for which response time requirements are met. In this case requirements can be renegotiated, for example by suggesting minimal possible service response time $d_k^*(req_l)$:

$$d_k^*(req_l) = \min_k \{d_k(req_l)\}. \tag{21}$$

When required service path s_{k^*} is found (by solving either task (20) or (21)) in order to be able to guarantee requested service response time, resources on service path s_{k^*} have to be reserved.

4.3. Average service response time guaranties (DiffServ)

Assume, that each incoming service requests req_l belongs to certain class c_l ($c_l = 1, \ldots, C$). Each class c ($c = 1, \ldots, C$) is characterized by probability q_c, that response time requirements of requests from this class are met:

$$P\{d_l \leq d_l^*\} = q_{c_l}, \tag{22}$$

where d_l and d_l^* are respectively: request req_l response time and request req_l response time requirement.

The aim of the task of delivering average service response time guaranties is to assign each incoming service request req_l to such a service path s_{k^*} for which equation (22) holds. Since probability $P\{d_l \leq d_l^*\}$ for each service path s_k can be calculated by means of cumulative distribution function $F_{rk}(d_l)$ (see eq. (12)), the task of delivering average service response time guaranties can be formulated as follows:

$$k^* = \max_k \{k \in \{1, \ldots, K\} : F_{rk}(d_l) \leq d_l^*\}. \tag{23}$$

Similarly to the task of delivering strict guaranties, it is possible that neither of service paths allow to obtain required probability of meeting response time requirements. In such a case service path with the highest probability of meeting response time requirements may be suggested:

$$k^* = \min_k \{F_{rk}(d_l)\}. \tag{24}$$

5. Simulation study

In order to illustrate presented tasks of resource allocation and evaluate performance of proposed algorithms simulation study was carried out. In simulations an example system was set up. It consisted of three serially ordered atomic services ($m = 3$), each having three different versions ($n_1 = n_2 = n_3 = 3$).

In the system example it is assumed, that there are three distinguished request classes, each of which has predefined method of quality of service assurance. First request class is served according to *best effort* model (described in section 4.1.) in which average service request response time is minimized. Requests from second class are served according do *IntServ* model (section 4.2.) which allows to deliver strict guaranties for maximal response time. Requests from the third class are served according to *DiffServ* model (section 4.3.) in which guaranties on average response time are delivered. Third request class consists of four subclasses, each characterized by different probability ($q_1 = 0,8, q_2 = 0,7, q_3 = 0,6, q_4 = 0,5$) of meeting required service response time $d^* = 0,5s$.

System was fed with a Poisson stream of requests with average stream intensity $\lambda_0 = 50$. The share of each request class in overall stream was as follows: best effort - 50%, IntServ - 10% and DiffServ - 40%. Each subclass of DiffServ requests had 10% share in overall stream. The ratio of amount of requests from different requests classes was chosen to be similar to the ratio of traffic volume in real computer communication networks.

The aim of simulation was to evaluate performance of proposed resource allocation algorithms measured as response time guaranties delivered to distinguished traffic classes for increasing value of request arrival intensity. Exemplary results of performed simulations are presented on figures 4 and 5.

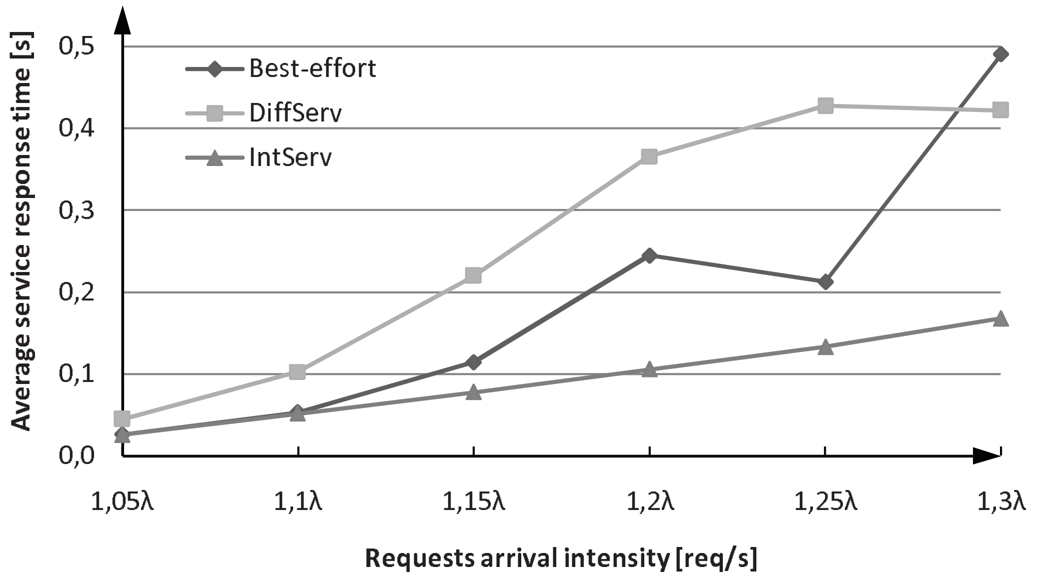


Fig. 4: Influence of increasing request arrival intensity λ on average service response time for three main request classes: best effort, IntServ, DiffServ.

On figure 4 the influence of increasing request arrival intensity λ on average service response time for three main request classes are presented. Requests from both best effort and IntServ classes are assigned resources such that average response time is minimized. There are two main differences between these two classes, namely resources for IntServ requests are reserved, what allows to provide strict guaranties on service response times. Moreover, IntServ requests have higher priority then best effort requests. In fact best effort requests priority is the lowest among all classes, therefore requests scheduling algorithms in atomic services assign to best effort requests only such amount of computational resources which are not consumed by other classes.

It is obvious, that for increasing request arrival intensity average service response time should increase as well for all request classes. An interesting situation occurs when request intensity reaches $\lambda = 1,25\lambda_0$. Average response time of requests from DiffServ class approaches its requirement $d^* = 0,5s$ and stops increasing. At the same moment response time of best effort request starts to decrease and afterwards rapidly increases. This is caused by the fact, that when DiffServ class reached its requirement it did not need as much resources as earlier. Excess resources were assigned to best effort class, what resulted in decreased response time. When request intensity increased DiffServ class needed more resources to provide required response time guaranties. Necessary resources were taken from best effort class, what caused rapid growth of best effort average response time.

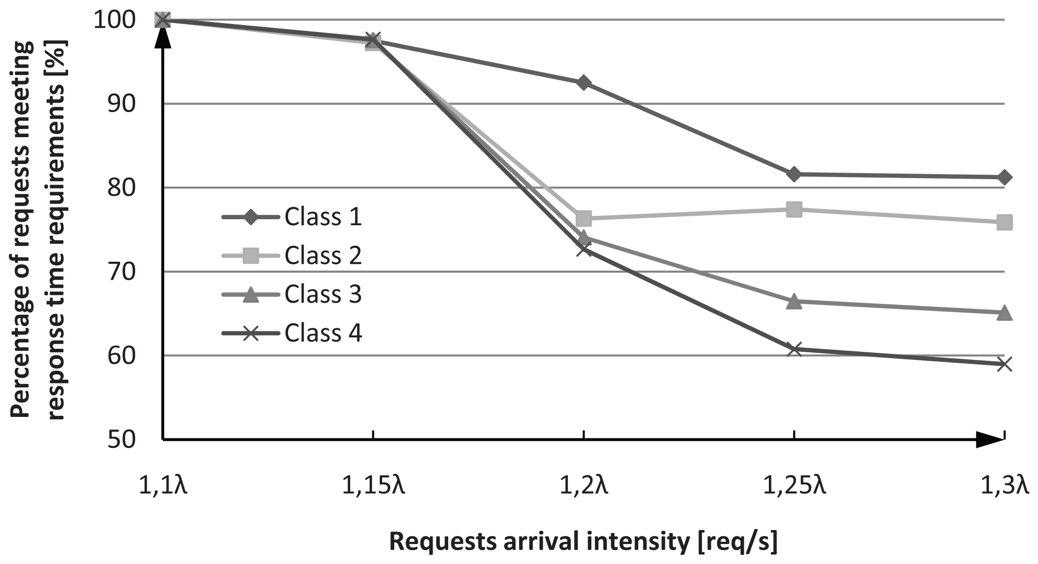


Fig. 5: Increasing request arrival intensity on the percentage of requests from each subclass of DiffServ meeting response time requirements.

Each subclass of DiffServ class have different requirements on percentage of requests meeting response time requirements. Exemplary results of quantitative analysis of the influence of increasing request arrival intensity on the percentage of requests from each subclass of DiffServ meeting response time requirements is presented on figure 5. One can notice, that as request arrival rate grows percentage of requests not violating response time guaranties approaches required values, which in presented study were set to $q_1 = 0,8, q_2 = 0,7, q_3 = 0,6, q_4 = 0,5$ for corresponding subclasses.

6. Final remarks

Research presented in this paper shows, that it is possible to deliver required level of quality of service and differentiate it between distinguished request classes by application of commonly known quality of service assurance approaches. It is worth noting, that presented resource allocation algorithms utilize only few methods (resource reservation, request scheduling) from classical QoS assurance models. Application of all QoS mechanisms (e.g.: traffic shaping and conditioning, request classification, contract renegotiation, congestion control, etc.) as well as knowledge engineering methods [5] (e.g.: prediction of client behavior, adaptive scheduling, atomic services load prediction, etc.) to management of systems resources may allow to significantly improve delivered quality of service.

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] S. Anderson, A. Grau, C. Hughes: Specification and satisfaction of SLAs in service oriented architectures, *5th Annual DIRC Research Conference*, pp. 141–150, 2005.

[2] N. Chee-Hock, S. Boon-Hee: *Queueing modelling fundamentals with application in Communication Networks, 2nd Edition*, Wiley and Sons, 2008, England.

[3] A. Grzech: *Teletraffic control in the computer communication networks*, Wroclaw University of Technology Publishing House, 2002, Wroclaw (in Polish).

[4] A. Grzech, P. Świątek: Modeling and optimization of complex services in service-based systems, *Cybernetics and Systems* , 40(08), pp. 706–723, 2009.

[5] A. Grzech, P. Świątek: Parallel processing of connection streams in nodes of packet-switched computer communication networks. *Cybernetics and Systems*, 39(2) pp. 155–170, 2008.

[6] N. Milanovic, M. Malek: Current Solutions for Web Service Composition, *IEEE Internet Computing* 8(6), pp.51–59, 2004.

[7] R. Rajan, D. Verma, S. Kamat, E. Felstaine, S. Herzog: A policy framework for Integrated and Differentiated Services in the Internet, *IEEE Network*, pp. 34–41, 1999.

[8] Z. Wang: *Internet QoS: architecture and mechanisms for Quality of Service*, Academic Press, 2001.