[EPR+07] *Julian Eckert, Krishna Pandit, Nicolas Repp, Rainer Berbner, Ralf Steinmetz*; **Worst-Case Performance Analysis of Web Service Workflows**. In: 9th International Conference on Information Integration and Web-based Application & Services (IIWAS 2007), Jakarta, Indonesia, December 2007. S.

Worst-Case Performance Analysis of Web Service Workflows

Julian Eckert, Krishna Pandit, Nicolas Repp
Rainer Berbner, Ralf Steinmetz[1]

Abstract

One major challenge in business process intelligence and business process management is performance modeling and performance measurement of workflows in order to plan their execution. Business process intelligence facilitates advanced business process execution management, i.e., prediction, monitoring, and optimization. In order to analyze Web Service workflows and to plan the workflow control, network calculus, originally developed for analyzing packet switched networks, can be used to describe the worst-case performance behavior of a workflow. By addressing capacity planning of Web Service workflows, resource usage becomes more and more important. Performance modeling and measurement are crucial to ensure that the workflow execution remains feasible and SLA violations due to overload are avoided. Thus, this paper presents a worst-case performance modeling approach for Web Service workflows based on network calculus to support capacity planning decisions.

1. Introduction

In the last years it became important for enterprises to react quickly to the changing environment and to adapt their business processes continuously due to the globalization and deregulation of markets. The realization of a continuous business process management with business process intelligence which allows to react flexibly and to manage the business process in order to avoid performance problems during the process execution is quite important. The IT architecture within enterprises is often characterized by a large amount of heterogeneous legacy systems, middleware platforms, programming languages, operating systems, and communication channels which are barely manageable [13]. Business process management has to meet customer expectations, i.e., Quality of Service (QoS), and has to control the costs to stay competitive [2]. Thus, enterprises have to plan their business processes in advance to adapt them to changing business needs.

Nowadays, it becomes more and more important to integrate internal legacy systems and to couple external business partners in order to realize flexible business processes. A Service-oriented Architecture (SOA) as an architectural blueprint, which facilitates the integration of internal legacy systems and the coupling of external business partners in order to realize flexible business processes can be used to support these goals [12]. By applying the SOA paradigm it is possible to compose and orchestrate self-contained loosely coupled services to cross-organizational business processes

[1] Multimedia Communications Lab, Department of Electrical Engineering and Information Technology, Technische Universität Darmstadt, Merckstr. 25, 64283 Darmstadt, Germany, {eckert, pandit, repp, berbner, steinmetz}@kom.tu-darmstadt.de

even from external partners. In order to implement workflows based on services, regardless of the underlying legacy systems, Web Services as an open standard are often used [3].

In our previous work [5] [6] [7] we presented a heuristic-based detailed workflow execution approach based on Web Service technology which enables the selection of specific Web Services at runtime as well as replanning mechanisms. In [8] a capacity planning approach for Web Service workflows is described in order to plan the workflow control in advance. To avoid the risk of poor workflow performance, business process intelligence, capacity planning of workflows, business process automation, and performance analysis are similarly important. Modeling workflows has been widely studied, e.g., in [1]. Generally statistical models are employed. However, they are not always appropriate. Certain customers demand deterministic QoS and are willing to pay for it. Further, the drawback of statistical models is that if the performance goal is not achieved, the question arises whether this is due to false assumptions or a statistical fluctuation.

In this paper, we focus on the worst-case performance analysis for Web Service workflows. The performance of a process is quite important for the workflow controller in order to decide how many instances of a workflow have to be executed to ensure that all requests can be served. We use network calculus as the tool for modeling the worst-case performance.

2. Network Calculus Basics

Network calculus is a system theory for deterministic queuing systems. It has been developed in the 1990s and is widely used in the context of deterministic QoS in packet switched networks. The system theory approach generally means modeling a phenomenon by three entities: a system, an input and an output. The system maps an input to an output. This approach is widely spread in the field of electrical engineering, especially in communications and control systems. Network calculus brings the system theory approach to computer networks. The input is the traffic flow that is originated at the sender, the system models the network which introduces delays, packet loss, etc., and the output is the traffic flow that arrives at the receiver. With this model, the performance of a network can be analyzed by evaluating whether the traffic flow arriving at the receiver is appropriate. One particularity about network calculus is that in contrast to classical system theory, it builds on min-plus algebra as a mathematical foundation. In min-plus algebra the addition operator is the "minimum"-operator and the multiplication operator is the "plus"-operator. The reason for this is that while normal, i.e., "plus-times" algebra is well suited for describing physical systems, min-plus algebra is suited for describing man-made systems [4]. The characteristic of a man-made system generally is that there are customers which share a resource while having a goal. In computer networks, the customers are data packets, the resource is the network and the goal is to maximize the throughput or to minimize the delay.

We will show in this paper that this model is also suited to describe business processes, where the customers are the requests, the shared resources are the service providers and the goal is something along the lines of maximizing the number of executed requests, minimizing the cost, etc.

This chapter provides a non-mathematical overview of the concepts and definitions used in the course of this paper. An excellent textbook on network calculus is [9], a concise introduction covering all theorems and definitions used here can be found in [10].

2.1. Arrival Curve

The arrival curve is the input to the system. Unlike in classical system theory, where the input is the actual signal that enters into the system, the arrival curve is the upper bound for the input traffic, i.e., the arrival curve denotes the maximum traffic that the sender may inject into the network. It is given as a function, where the x-axis denotes the time interval and the y-axis denotes the maximum amount of data that may be sent in the corresponding interval. An example of an arrival curve, along with the other concepts presented in this section, is shown in Figure 1. The most prominent example for a traffic regulation algorithm is the Leaky Bucket [15], which is often also referred to as Token Bucket. Its arrival curve is given by the following equation.

$$a(t) = b + rt \text{ for } t > 0 . \quad (1)$$

It indicates that a burst of size b can be sent at once, but thereafter only the rate r can be sent.

2.2. Service Curve

The service curve indicates the amount of data that a node must serve in the worst case. It is also given in the form of a function, where the x-axis denotes a time interval and the y-axis the minimum amount of data that must be served. A widespread service curve is the latency-rate (LR) service curve [9] [10], which has two parameters. After the latency the packets are served at a constant rate. In the following we introduce two of three basic bounds of Network Calculus, the backlog bound and the delay bound. In addition the concatenation of service curves is shown which provides a method to represent several network elements as one system

2.3. Backlog Bound

The backlog bound denotes the amount of data that is in the network. If only one node is considered, the backlog bound denotes the minimum buffer size of that node. The backlog bound is the maximum vertical distance of the arrival curve and service curve.

2.4. Delay Bound

The delay bound denotes the maximum delay that a packet might experience in the network. It is given by the horizontal distance of the arrival curve and service curve. An example of an arrival curve and service curve with the corresponding backlog and delay bound is shown in Figure 1.

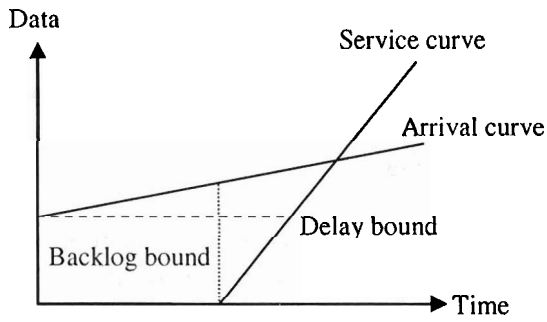

Figure 1. Arrival curve, service curve, backlog bound, and delay bound

2.5. Concatenation

The concatenation theorem allows the analysis of networks consisting of multiple nodes. It works analogous to classical system theory, where the transfer function of a system is given by the convolution of the transfer functions of the subsystems. The service curve of a path is given by the min-plus convolution of the service curves of the nodes along the path. The min-plus convolution is given by the following function, where $\beta_1(t)$ and $\beta_2(t)$ are the node service curves and $\beta(t)$ is the service curve of the path consisting of these two nodes.

$$\beta(t) = \min_{0<\tau\leq t} \{\beta_1(\tau) + \beta_2(t-\tau)\}. \quad (2)$$

For a method how to intuitively convolve arbitrary functions the reader is referred to [11] (elaborated in [10]).

3. System Model

Typically a business process is defined as a consecution of activities which creates a value to the customer. The automation of a business process by IT-systems is referred to as a workflow [16]. Assuming a business process which can be decomposed into several basic activities, each of these activities can be executed by a Web Service WS_i $(i=1,...,n)$. For details on the selection and invocation of a Web Service, the reader is referred to [17]. The selected and invoked Web Services can be composed to a Web Service workflow, which facilitates the selection and inter-connection of Web Services provided by different service providers or partners [17].

In the following we will analyze sequential Web Service workflows, i.e., each Web Service WS_i is executed sequentially. The workflow can be decomposed into several subprocesses. Depending on the granularity of the activities, each subprocess can be further decomposed into subtasks. Considering a workflow consisting of n different activities the workflow controller has to ensure that task i $(i=1,...,n)$ has to be executed before task i' $(i'=1,...,n)$ if $i<i'$. For each task the workflow controller has to select the appropriate Web Service that provides the required functionality for the specific task i $(i=1,...,n)$ and has to create a detailed execution plan for determining which Web Service has to be invoked at which step in the workflow. For each activity there exist several Web Services which fulfill the required functionality of the task. The workflow controller has to create the execution plan, i.e., to select those Web Services, which are the most cost-efficient. Furthermore, it has to reduce the delay, maximize the throughput, and realize an optimal resource usage. In the considered system, the incoming requests for the execution of the workflow can arrive in a bulk at a specific time or can arrive constantly.

4. Applying Network Calculus to Web Service Workflows

4.1. Arrival Curves and Service Curves

As pointed out in the system model, requests to the system can arrive in different forms. Possibilities are bulk arrivals, constant rate arrivals or stochastic arrivals. Since our goal is to describe the deterministic worst-case behavior, we neglect purely stochastic arrival models, as they might cause the worst-case performance to be arbitrary bad, even if the probability for that case is low. The Token Bucket arrival curve is appropriate to capture the incoming request arrivals. It allows for setting the maximum size of the bulk of arrivals as well as the sustained rate.

The relevant parameters for our scenario to describe a Web Service are the response time and the rate at which requests can be serviced. Therefore, the service curve concept is well-suited to describe the performance of a Web Service. The latency-rate (LR) service curve [9] [10] grasps exactly the two aforementioned parameters. Beyond that, it is also possible to have service providers, which offer Web Services in the form of "the first 100 request are serviced with a rate 10 executions/second and the remaining ones with rate 5 executions/second". This is grasped by the L2R service curve [14], which will be neglected in our further analysis. Using sophisticated service curves allows the service provider to do cost differentiation. When determining the appropriate parameters of the service curve the effect of other workflows being executed at the server, possibly leading to congestion, must be incorporated as well.

As mentioned earlier, the examined workflow consists of sequentially executed Web Services. Hence, if a Web Service is described by a service curve, the workflow can be described as the concatenation of the service curves. This is an analogy to a network path, where the service curve of the path consists of the concatenation of the service curves of the nodes. In Table 1 the analogies of a packet switched network and a Workflow Management System (WFMS) facilitating Web Services for implementation purposes are summarized.

Packet switched network	Workflow Management System
Path through the network	Workflow
Node in the network	Web Service
Packet	Request
Throughput	Rate at which requests can be processed
End-to-end delay	Time until a workflow is completed

Table 1. Analogies between a packet switched network and the considered WFMS

4.2. Application of the Theorems to Web Services

In this section we interpret the theorems on the bounds in the context of Web Services and Web Service workflows, respectively. The delay bound indicates the worst-case response time that a request may have. It takes place when the largest possible bulk of requests arrives and the service provider offers exactly the service curve. This is shown in Figure 7. From the backlog bound it can be deduced how many requests are currently pending, which equals the amount of requests not yet executed.

4.3. Optimal Choice of Service Providers

In order to use the best suited Web Services to realize the execution of the workflow, the workflow controller has to create the overall service curves for the workflow. Later it has to choose those Web Services with which it is able to maximize his throughput and to minimize the worst-case delay.

An example of such a workflow is shown in Figure 2. The incoming requests are described by the arrival curve $a(t)$, the worst-case of the processing of each Web Service WS_i is specified with $\beta_i(t)$. The worst-case overall service curve of the workflow is depicted as $\beta_{j,workflow}(t)$. It must be noted that $a(t)$ is not the actual input of the system and $\beta_{j,workflow}(t)$ does not describe the actual scheduling. They are bounds for the input and the scheduler.

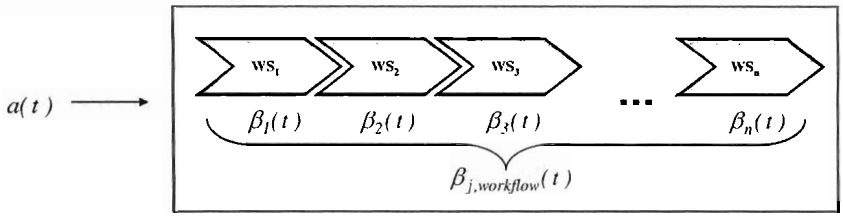

Figure 2. Web Service workflow

The results of [11] (elaborated in [10]) can be used to determine the optimal combination of service providers with respect to a certain goal. Minimizing the worst-case response time of a first request in a bulk is achieved by choosing for each task the Web Service with the service curve having the lowest latency and the highest execution rate.

The throughput is maximized as follows. For each Web Service the service curve with the highest rate is collected. Theses rates are compared, and the lowest of them is the highest achievable rate for the workflow. Since it does not help if other service providers offer higher rates than the highest achievable rate, from them service curves offering the highest achievable rate can be chosen. Optimizing to a joint rate and delay requirement is done by selecting from each service provider the service curve with the lowest delay under the condition that the rate requirement is met.

5. Example

Considering a generic credit process, the appropriate Web Service workflow has to be executed many times in a specific time period to serve all incoming requests. Assuming that the incoming requests, arriving at the Workflow Management System, are constrained by a Token Bucket arrival curve $a(t)$ with the rate r and a depth b.

Each Web Service offers a functionality, i.e., the execution of the requested service, to the service consumer. The amount of requests a Web Service WS_i is able to serve within a specific time slot – which is specified respectively negotiated with the service provider in the SLAs – can be modeled as a worst-case with a service curve $\beta_i(t)$. Figure 3 shows our example process, which will be analyzed in the following. The credit process can be decomposed into the subprocesses loan request, credit assessment, servicing, and workout.

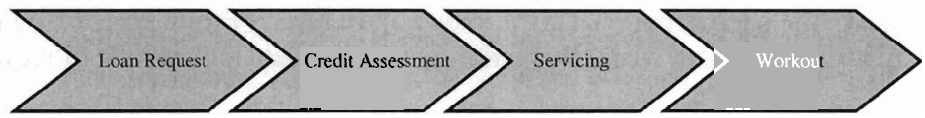

Figure 3. Generic Credit Process

In order to model the incoming requests to execute the credit process, we will use two Token Bucket arrival curves in our example, as depicted in Figure 4.

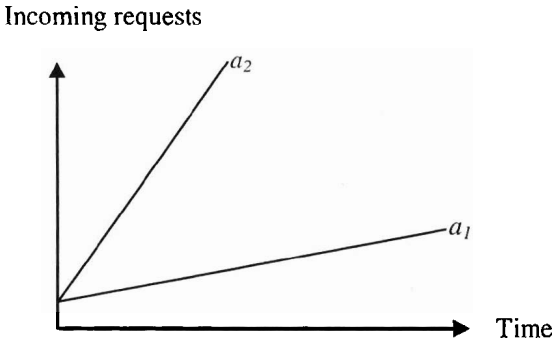

Figure 4. Arrival curves for incoming execution requests

For the execution of each task, a Web Service can be used which is offered by a service provider. Furthermore there exist service providers offering several Web Services which fulfill the required functionality for each task, as e.g. the loan request. In order to reduce the complexity of the example it is assumed that there are three Web Services available for the first task and two Web Services for each of the remaining tasks. In the following, the Web Services for the first task are denoted as $WS_{1,1}$, $WS_{1,2}$, $WS_{1,3}$ where the first index denotes the activity and the second index the number of the considered Web Service, which fulfills the required functionality of this task. The Web Services for the second task are denoted as $WS_{2,1}$, $WS_{2,2}$ and so on.

Each Web Service offers different service levels. A Web Service needs some time for initialization (latency l) and thereafter begins with the execution of requests with a specific rate r. The differences between the offered Web Services are on the one hand the time needed for initialization, i.e., the latency l, and on the other hand the execution rate r. This behavior can be described with rate latency service curves as shown for the first task in Figure 5.

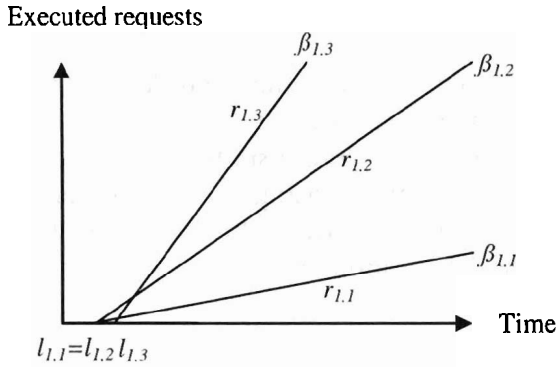

Figure 5. Service curves for the first task

In this example the execution of $WS_{1,1}$ starts with a small latency $l_{1,1}$ and with a slow execution rate $r_{1,1}$. $WS_{1,2}$ has the same latency of $l_{1,2}$ but a higher execution rate $r_{1,2}$. The highest execution rate $r_{1,3}$ with the largest latency of $l_{1,3}$ has $WS_{1,3}$. Figure 6 describes the service curves for the offered Web Services for the remaining tasks. The challenge for the workflow controller will be to invoke those Web Services in order to achieve the lowest delay respective the highest throughput. Thus, with the introduced concatenation theorem it is able to construct the overall service curves for the process for the different chosen Web Services and can optimize his throughput and minimize the delay.

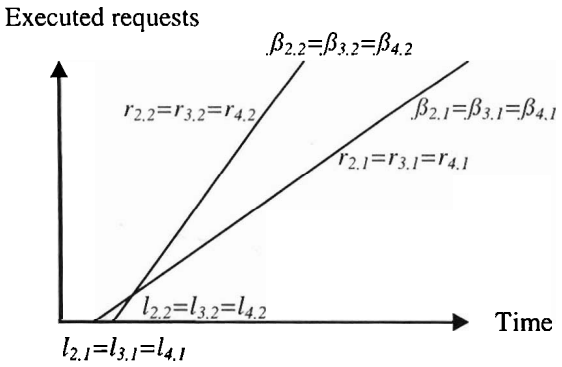

Figure 6. Service curves for the remaining tasks

5.1. Delay Analysis

If the workflow controller chooses $WS_{1,1}$, $WS_{2,1}$, $WS_{3,1,}$, and $WS_{4,1}$ to execute the process, the overall service curve results in $\beta_{1,workflow}$ with the delay d_1 shown in Figure 7.

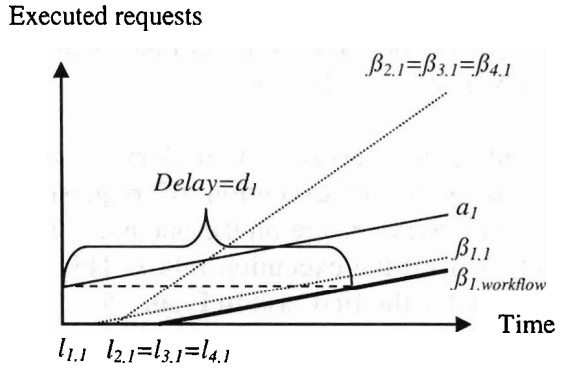

Figure 7. Delay for an aggregated service curve

The maximum delay that a request will take to complete is denoted by the dashed line. This delay d_1 sets in for the last request that arrives in the largest possible bulk allowed by the arrival curve and the service provider waiting its full latency before starting to service requests. It is also to be noted that the throughput could be increased by choosing the service curve $\beta_{1,2}$, but it is not required in this case as the rate of the arrival can be serviced this way as well. However, the delay could be reduced when using $\beta_{1,2}$. In these deliberations it becomes clear that optimizing the choice of Web Services is a non-trivial problem.

5.2. Throughput Analysis

In order to optimize the throughput of the process, the workflow controller has to choose the Web Services with the highest execution rates. In our example this would comply with $WS_{1,3}$, $WS_{2,2}$, $WS_{3,2,}$, and $WS_{4,2}$. The overall service curve results in $\beta_{2,workflow}$ which is shown in Figure 8. In this case it becomes clear, that even though the rate is higher, the latency prior to the first request processed is higher than in the previous example.

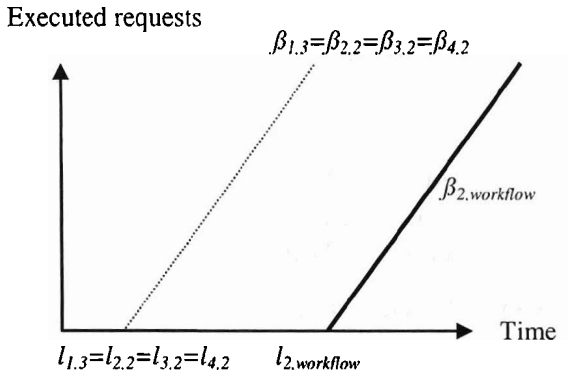

Figure 8. Throughput Maximization

6. Conclusion

In this paper we propose a worst-case performance model for Web Service workflows. The rationale of this model is to apply results from network calculus to Web Service workflows. The concept of arrival curves is used to describe the arrivals of requests. The concept of service curves is well-suited to capture the service offered by a service provider. Further, service curves allow a flexible description of the services offered by the service provider, which is useful when charging is taken into account. It is shown how the delay and the throughput of a workflow execution can be optimized.

We understand that the sequential workflows considered in this paper are only a subset of the workflows encountered in reality. However, network calculus allows the analysis of complex networks (and therefore complex workflows) containing complex arrival and service patterns, loops, forking and joining paths, etc. [4] [9]. Incorporating these concepts in the analysis of workflows is subject of our future work.

Beyond that, our further research aims at extending our approach with a cost model and formulating an optimization problem. Further, we will enhance the approach considering workflow parallelization and other resource planning problems and implement our results in a prototype.

Acknowledgments

This work is supported in part by the E-Finance Lab e.V., Frankfurt am Main, Germany (http://www.efinancelab.com).

References

[1] Aalst, W. M. P. v. d., Hee, K. M. v.: Workflow Management: Models, Methods, and Systems. MIT press, Cambridge MA (2002).

[2] Almeida, V. A. F., Menascé, D. A.: Capacity Planning: An Essential Tool for Managing Web Services. In: IT Professional 4:4 (2002) 33-38.

[3] Alonso, G., Casati, F., Kuno, H., Machiraju, V.: Web Services. Concepts, Architectures and Applications. Springer-Verlag, Berlin Heidelberg New York (2004).

[4] Baccelli, F.; Cohen, G.; Olsder, G.-J.; Quadrat, J.-P.: Synchronization and Linearity: An Algebra for Discrete Event Systems. John Wiley and Sons (1992).

[5] Berbner, R., Grollius, T., Repp, N., Heckmann, O., Ortner, E., Steinmetz, R.: An approach for the Management of Service-oriented Architecture (SoA) based Application Systems. In: Enterprise Modelling and Information Systems Architectures (EMISA 2005). Klagenfurt Austria (2005).

[6] Berbner, R., Heckmann, O., Steinmetz, R.: An Architecture for a QoS driven composition of Web Service based Workflows. In: Networking and Electronic Commerce Research Conference (NAEC 2005). Riva del Garda Italy (2005).

[7] Berbner, R., Spahn, M., Repp, N., Heckmann, O., Steinmetz, R.: Dynamic Replanning of Web Service Workflows. In: IEEE International Conference on Digital Ecosystems and Technologies 2007 (IEEE DEST 2007). Cairns Australia (2007).

[8] Eckert, J., Repp, N, Schulte, S., Berbner, R., Steinmetz, R.: An Approach for Capacity Planning for Web Service Workflows. In: 13th Americas Conference on Information Systems (AMCIS 2007). Keystone Colorado USA (2007).

[9] Le Boudec, J.-Y., Thiran, P.: Network Calculus. Number 2050 in Lecture Notes in Computer Science. Springer-Verlag, Berlin Heidelberg New York (2001).

[10] Pandit, K.: Quality of Service Performance Analysis based on Network Calculus. PhD Thesis, Dept. of Electrical Engineering, Technische Universitaet Darmstadt (2006).

[11] Pandit, K., Schmitt, J., Kirchner, C., Steinmetz, R.: A Transform for Network Calculus and its Application to Multimedia Networking. In: SPIE Conference on Multimedia Computing and Networking (MMCN 2006). San Jose (2006).

[12] Papazoglou, M. P.: Service-Oriented Computing: Concepts, Characteristics and Directions. In: 4th International Conference on Web Information Systems Engineering (WISE 2003). Rome Italy (2003) 3-12.

[13] Papazoglou, M. P., van den Heuvel, W. J.: Leveraging legacy assets. In: Papazoglou, M. P., Spaccapietra, S. and Tari, Z. (eds.): Advances in Object-Oriented Modeling. MIT Press, Cambridge MA (2000) 131-160.

[14] Schmitt, J.: On the Allocation of Network Service Curves for Bandwidth/Delay-Decoupled Scheduling Disciplines. In: Proceedings of the IEEE Global Telecommunications Conference 2002 (GLOBECOM 2002). Taipei Taiwan (2002).

[15] Turner, J.: New Directions in Communications (or Which Way to the Information Age?). In: IEEE Communications Magazine 24:10 (1986) 8-15.

[16] Workflow Management Coalition: Workflow Management Coalition – Terminology & Glossary. Technical Report WFMC-TC-1011, Hampshire United Kingdom (1999).

[17] Zeng, L., Benatallah, B., Ngu, A., Dumas, M., Kalagnanam, J., Chang, H.: QoS-aware Middleware for Web Service composition. In: IEEE Transactions on Software Engineering 30:5 (2004) 311-328.