Julian Eckert, Tim Lehrig, Apostolos Papageorgiou, Nicolas Repp, Ralf Steinmetz:
Solving Resource Planning Problems – A Heuristical Solution. In: 11th International Conference on Information Integration and Web-based Application and Services (iiWAS 2009), p. 348-353, ACM, December 2009. ISBN 978-3-85403-260-1.

Solving Resource Planning Problems – A Heuristical Solution

Julian Eckert
Technische Universität Darmstadt
Multimedia Communications Lab - KOM
Merckstr. 25, 64283 Darmstadt, Germany
Julian.Eckert@kom.tu-darmstadt.de

Tim Lehrig
Technische Universität Darmstadt
Multimedia Communications Lab - KOM
Merckstr. 25, 64283 Darmstadt, Germany
Tim.Lehrig@kom.tu-darmstadt.de

Apostolos Papageorgiou
Technische Universität Darmstadt
Multimedia Communications Lab - KOM
Merckstr. 25, 64283 Darmstadt, Germany
Apostolos.Papageorgiou@kom.tu-darmstadt.de

Nicolas Repp
Technische Universität Darmstadt
Multimedia Communications Lab - KOM
Merckstr. 25, 64283 Darmstadt, Germany
Nicolas.Repp@kom.tu-darmstadt.de

Ralf Steinmetz
Technische Universität Darmstadt
Multimedia Communications Lab - KOM
Merckstr. 25, 64283 Darmstadt, Germany
Ralf.Steinmetz@kom.tu-darmstadt.de

ABSTRACT

Resource planning for service-based workflows becomes crucial considering a large amount of workflow execution requesters in a SOA or Grid environment. Especially, business process management and performance management of service-based workflows are of high importance avoiding performance degradation. The need for efficient resource planning techniques forces intermediaries, acting as workflow orchestrators, to use efficient heuristics for the determination of service invocation plans for workflows at short computation times. This paper presents an optimization approach for the resource planning problem and proposes an efficient heuristical solution solving the addressed optimization problem at a high solution quality and at a short computation time.

Keywords

Distributed Workflows, Resource Planning, Service-oriented Computing, Service Composition, Quality of Service

1. INTRODUCTION

The globalization and the growing industrialization led enterprises to become very agile and flexible. In particular, business processes and workflows have to become very flexible and have to be adaptable in various conditions. Due to the increasing competition, enterprises are forced to realize cost-efficient workflows that meet customer requirements. Beside the functional requirements of a business process, also the non-functional requirements are very important. Thus, the provision of a certain Quality of Service (QoS) level as well as an effective business process management is crucial in order to increase the market share of an enterprise.

Concerning cross-organizational workflows, i.e., workflows in which services are sourced and invoked from internal as well as from external partners to a single workflow, business process management is necessary for reliable operations [9]. The on-demand integration of external, loosely coupled services as well as the integration of internal legacy systems is provided by the concept of a Service-oriented Architecture (SOA) [8]. Furthermore, SOA represents an approach facilitating the needed flexibility and feasibility in aligning application landscapes to business-driven demands [7].

The general scenario of this paper is the *Internet of Services* in which a large variety of services with specific functionalities exist that solely vary in their non-functional parameters such as response time, execution capacity, and costs [4]. These services are listed in central or decentral repositories. Furthermore, negotiation of Service Level Agreements (SLAs), service monitoring, service pricing, and service billing are supported. In particular, several roles at a high degree of SOA Maturity [5] are existent such as the *service provider*, the *service consumer*, the *service intermediary*, the *service executor*, the *platform host* as well as a *service marketplace*. The available services are offered on a big scale on a so-called service marketplace.

iiWAS2009, December 14–16, 2009, Kuala Lumpur, Malaysia.

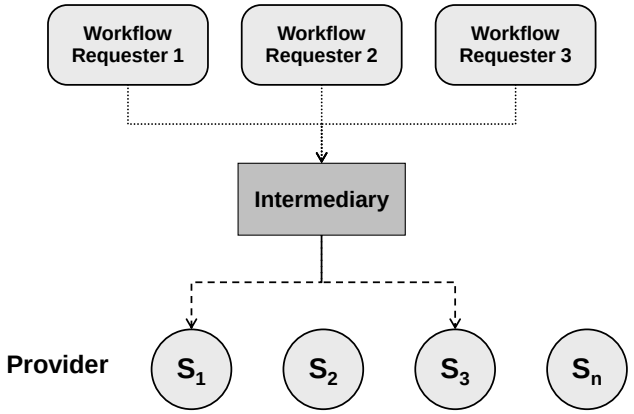

Figure 1: Research scenario

Focusing on cross-organizational service-based workflows, assuming a large set of workflow requesters and service providers, an intermediary is responsible for workflow composition as presented in Figure 1. Furthermore, the intermediary has to ensure that all workflow execution requests can be served at the demanded QoS levels. In particular, this paper focuses on the following roles with the associated tasks as described in the following:

- Service provider: Offers services with a specific functionality at varying QoS levels such as response time, service execution capacity, and specific costs (several cost models are possible).
- Workflow requester: Requests a specific workflow execution with specific QoS and cost requirements.
- Intermediary: Monitors, aggregates, and prioritizes all workflow execution requests and implements and applies a resource planning process [2].

The intermediary is responsible for the detailed resource planning of workflows, i.e., the selection and invocation of services taking into account constraints such as response time constraints or constraints concerning the workload (amount of workflow execution requests). Furthermore, the intermediary has to avoid the risk of poor workflow performance and SLA violations in order to ensure satisfaction of workflow consumers.

Thus, this paper analyzes the problem of resource allocation for an intermediary. The problem of multiple parallel service selections and invocations is formulated as an optimization problem that is proven to be NP-hard. Furthermore, this paper provides an efficient heuristic in order to solve this problem in real-time, i.e., at low computation time with a high solution quality. An overview about related work in the field of resource planning of service-based workflows is given in [2].

The remainder of this paper is structured as follows. After an overview about the system model, described in section 2, the optimization model of the intermediary is presented in section 3. Furthermore, the developed heuristical solution is presented in section 4 with an extensive evaluation in section 5. The paper closes with a summary and an outlook on future work.

2. SYSTEM MODEL

As mentioned previously, this paper focuses on cross-organizational workflows with a high repetition rate. In particular, the considered workflow can be decomposed into several basic activities, which can be executed by specific services S_i $(i = 1, \ldots, n)$ fulfilling the required functionalities of the considered tasks. These tasks are, depending on the granularity of the task decomposition, decomposed in a way that there exist ranges of services $S_{i,j}$ $(j = 1, \ldots, m_i)$ which fulfill the required functionality of the considered task i. As the intermediary is responsible for the execution of a workflow, he has to handle all workflow execution requests and has to ensure that all requests can be served within a specific time period. Furthermore, the objective of the intermediary is to serve all incoming workflow execution requests at minimal costs and at demanded QoS properties such as response time. It is assumed that the services have a limited execution capacity $cap_{i,j}$ and that for workflow execution, several services have to be invoked (also in parallel if necessary) and composed to a workflow. Appropriate services, which fulfill the required functionality, described in the SLAs, have to be selected for each task. In the considered workflow consisting of n different activities the intermediary has to ensure that task $i(i = 1, \ldots, n)$ has to be executed before task $i'(i' = 1, \ldots, n)$ if $i < i'$. The intermediary has to create an invocation plan in order to use the execution capacities of each service optimally, i.e., to select the services in the most efficient way.

The intermediary aggregates all workflow execution requests, prioritizes them and determines the workload W that has to be executed within a specific deadline τ. Furthermore, the selected services have to be able to serve this workload at demanded QoS properties. The challenge will be to determine cost-efficient invocation plans, which fulfill the QoS demands and guarantee the feasible execution of all workflow execution requests.

3. OPTIMIZATION MODEL

As the composition of services to a workflow can be regarded as an optimization problem, this section focuses on the formulation of the analyzed optimization model. Several QoS properties as the response time of a service $t_{i,j}$, the limited execution capacity $cap_{i,j}$, and the costs for service invocation $c_{i,j}$ are of high importance. Concerning the pricing model, a volume-oriented pricing is assumed in this scenario, i.e., the costs $c_{i,j}$ are charged to the intermediary for the execution of up to $|cap_{i,j}|$ service executions [3]. As it is the objective to minimize the service invocation costs, Equation 1 formulates the objective function of the optimization problem.

$$MinF(\vec{x}) = \sum_{i=1}^{n} \sum_{j=1}^{m_i} c_{i,j} * x_{i,j} \quad (1)$$

Beside this objective, the intermediary has to ensure that the invoked services are able to serve all incoming workflow execution requests W as depicted in Equation 2 and to

ensure that the service composition is able to execute all requests within a specific deadline τ as presented in Equation 3. As several services have to be invoked in parallel in some categories in order to serve all incoming workflow execution requests Equation 4 and 5 ensure that multiple service executions are possible. Furthermore, $x_{i,j}$, as a binary variable, indicates whether the service $S_{i,j}$ is chosen or not.

$$\sum_{j=1}^{m_i} cap_{i,j} * x_{i,j} \geq W \quad \forall i = 1, \ldots, n \tag{2}$$

$$\sum_{i=1}^{n} max\left\{t_{i,j} | x_{i,j} = 1\right\} \leq \tau \tag{3}$$

$$\sum_{j=1}^{m_i} x_{i,j} \geq 1 \quad \forall i = 1, ..., n \tag{4}$$

$$x_{i,j} \in \{0, 1\} \tag{5}$$

As presented in [2] due to the combinatorial nature a Multi-choice Multi-dimensional Knapsack Problem is proven to be NP-Hard [6]. In particular, it can be deduced that this presented optimization problem exceeds the complexity of a Multi-choice Multi-dimensional Knapsack Problem as multiple services can be chosen from each category. However, this serves as a prove for the NP-hard complexity of the problem and shows the importance of heuristical solutions. In the work of [2] a heuristical solution, $H1\_G.KOM$, has been developed that shows significant outperformance in terms of computation time and an excellent solution quality. Depending on varying problem sizes (varying number of process steps n and varying number of service candidates m_i) the problem was solved with $H1\_G.KOM$ and an exact solution, i.e., by the lp_solver[1] (using a simplex algorithm and Branch & Bound). The evaluations show a relative processing time of less than 0.4% of $H1\_G.KOM$ with a solution quality of about 95% on average.

In addition to this heuristical solution, another heuristic, $H2\_G.KOM$, is developed that reduces the presented optimization problem with the help of a search space reduction. Furthermore, as described in the following section, the relaxed problem is solved exactly and a backtracking solution approach is applied.

4. HEURISTICAL SOLUTION

In order to benchmark the developed heuristic $H1\_G.KOM$ with existing heuristics, an adapted heuristical solution has been developed referred to as $H2\_G.KOM$ using the following existing heuristical solutions $H1\_RELAX\_IP$, and $H2\_SWAP$ of [1]. Furthermore these heuristical solutions are adapted to the presented resource planning problem in Section 3. As a difference, in Equation 4 it is only allowed that one service per category can be selected. Thus, Equation 4 has to be replaced by Equation 6 ensuring that only one service per category i is selected.

$$\sum_{j=1}^{2^{m_i}} x_{i,j} = 1 \quad \forall i = 1, \ldots, n \tag{6}$$

[1]http://lpsolve.sourceforge.net/5.5/, Version 5.5.0.12

Furthermore, the existing heuristical approach has to be adapted in order to be able to handle multiple service invocations, i.e., the selection of more than one service in each category i. Thus, the following presents the procedure and the properties of the developed new heuristical solution $H2\_G.KOM$.

As in the resource planning problem multiple services per category can be selected, it is needed to build up aggregated services out of the services m_i in category i. For aggregation purposes, several aggregation operators can be used depending on the considered QoS parameter. In order to create all possible service combinations in one category, the power set $P(S)$ of alternative services is determined. After building the power set, each category contains 2^{m_i} service candidates that is taken into account in Equation 6 and must also be adapted in Equations 1 and 2. The considered non-functional parameters of the services such as response time, execution capacity, and costs have to be aggregated with the help of the additive and maximum operator for the category level aggregation. After building the power set, each category contains 2^{m_i} services that are listed in category list $cat\_list_i[]$ for each category i. In the following, the non-functional aggregated parameter values of the aggregated services are presented as $t'_{i,j}$, $c'_{i,j}$, $cap'_{i,j}$ whereas j ranges from 1 to 2^{m_i} in each category i after this transformation.

After this transformation, the problem is reduced to a Multi-choice Multi-dimensional Knapsack Problem and can be solved with existing heuristics. Due to the large computation time of those problems, it is beneficial to reduce the number of services per category further, taking into account that a large reduction may induce worse values of the objective function. Thus, the number of service candidates in $cat\_list_i[]$, is reduced during the aggregation phase with the help of the following procedure for each category i:

1. Check whether the aggregated service fulfills the capacity constraint, i.e., check whether $cap'_{i,j} \geq W$. If this is not the case, do not insert $S'_{i,j}$ in $cat\_list_i[]$.

2. Check whether the aggregated service has lower costs or takes less time than the aggregated service with the lowest costs so far, if not do not insert $S'_{i,j}$ in $cat\_list_i[]$.

3. Check whether the aggregated service has lower costs or takes less time than the aggregated service with the lowest time so far, if not do not insert $S'_{i,j}$ in $cat\_list_i[]$.

After the aggregated services have been aggregated and listed in $cat\_list_i[]$ this search space in terms of alternative services can be further reduced by the following algorithm:

1. Sort $cat\_list_i[]$ in ascending order concerning the costs $c'_{i,j}$.

2. Remove all services that have a higher or the same response time at a higher or the same cost level than any other service in category i, i.e., remove service if $t'_{i,j} \geq t'_{i,k} \wedge c'_{i,j} \geq c'_{i,k}$ for $k \neq j$.

These algorithms lead to a further reduction of the search space in order to increase the performance of the applied

heuristics. The number of services 2^{m_i} after the transformation in Equations 1 and 6 can be replaced by m'_i, whereas $m'_i \leq m_i$. Because of considering the characteristics of the aggregated values during the aggregation process the following argumentation for $m'_i \leq m_i$ can be stated. As the response time is aggregated by a maximum operator on category level, after the creation of the power set, only m different response time values exist in the set of services. Since costs follow an additive operator, the services with the same response time value solely differ in their costs. The service with the minimum cost value dominates the other services leading to a deletion of the remaining services with identical response times. This proves that the number of services after the reduction m'_i is $\leq m_i$. As all aggregated services have to fulfill the capacity constraint also less than m response time values can exist leading to a further decrease of m'_i. In addition, the complexity of the underlying problem is further reduced by eliminating Equation 2 by applying step 1 in the aggregation phase. In order to adapt the overall response time constraint to the transformation and reduction process, Equation 3 has to be replaced by Equation 7.

$$\sum_{i=1}^{n} \{t_{i,j} | x_{i,j} = 1\} \leq \tau \tag{7}$$

After the search space has been reduced and the problem has been simplified, the following heuristical solutions of [1] are applied on the resulting set of services:

1. Apply $H1\_RELAX\_IP$, i.e., solve the relaxed problem (integer variables are reduced to real variables) with the lp-solver[2] (Simplex algorithm) and determine a feasible solution with the help of the backtracking solution of [1].

2. Apply $H2\_SWAP$, i.e., swap randomly services in the solution with other services and try to improve the solution quality, i.e., the objective function.

As the heuristic $H2\_G.KOM$ provides a further solution for the resource planning problem, the next section shows an evaluation and compares those heuristical solutions.

5. EVALUATION

After the description of the heuristical solution, this section focuses on the evaluation of $H2\_G.KOM$. The solution of the developed heuristic is compared to an exact solution (integer programming approach). Especially, the computation time and the solution quality are analyzed with respect to influencing factors. All experiments were run on an Intel Core 2 Duo (1.86 GHz) system with 2 GB of RAM, running Windows XP.

The evaluation is originally based on 16 different problem sizes, i.e., a varying number of process steps i and varying number of service candidates m_i whereas the number of test cases is fixed to $m_i = m$. Several problem sizes have been determined and for test case generation several sets concerning the correlation between the two analyzed non-functional characteristics and the costs have been generated. The QoS-CoS correlation of the test cases is measured with the Pearsons correlation coefficient. For the presented extract of the evaluation the parameters of test case generation have been adjusted to a resulting correlation level of 55% whereas the costs and response time are negatively and the execution capacity and costs are positively correlated. The response time restriction strength is set to 100% and the execution capacity strength to 30%. Thus, 10 test sets with 25 test cases each and different correlation levels have been generated. In detail the following problem sizes $n|m$ have been generated: 4|4, 4|6, 4|8, 4|10, 6|4, 6|6, 6|8, 6|10, 8|4,..., 8|10, 10|4,..., 10|10.

[2]http://lpsolve.sourceforge.net/5.5/, Version 5.5.0.13

Res_Plan	Trans.	Reduced	Reduction Ratio
4|4	4|16	4|3	10^{-3}
4|6	4|64	4|3	10^{-6}
4|8	4|256	4|3	10^{-8}
4|10	4|1024	4|4	10^{-10}
6|4	6|16	6|2	10^{-6}
6|6	6|64	6|3	10^{-8}
6|8	6|256	6|4	10^{-11}
6|10	6|1024	6|5	10^{-14}
8|4	8|16	8|3	10^{-6}
8|6	8|64	8|3	10^{-11}
8|8	8|256	8|4	10^{-15}
8|10	8|1024	8|5	10^{-19}
10|4	10|16	10|2	10^{-10}
10|6	10|64	10|4	10^{-13}
10|8	10|256	10|5	10^{-18}
10|10	10|1024	10|5	10^{-24}

Table 1: $H2\_G.KOM$: Reduction of solution space

Problem Size	4|4	4|6	6|4	8|4
$H1\_G.KOM$	0.4977	0.4259	0.1058	0.0358
$H2\_G.KOM$	8.5103	2.9622	1.4687	0.2648

Table 2: Relative computation time in %, 1/3

Problem Size	4|8	6|6	4|10	10|4
$H1\_G.KOM$	0.0485	0.0271	0.0469	0.0048
$H2\_G.KOM$	0.5679	0.1121	0.5100	0.0310

Table 3: Relative computation time in %, 2/3

Problem Size	8|6	6|8	6|10	10|6
$H1\_G.KOM$	0.0016	0.0020	0.0009	0.0001
$H2\_G.KOM$	0.0060	0.0086	0.0079	0.0003

Table 4: Relative computation time in %, 3/3

Before starting with the detailed evaluation of the developed heuristic $H2\_G.KOM$ the search space reduction is analyzed. Table 1 gives an overview about the degree of the search space reduction. The first column presents the initial problem size that is analyzed in the resource planning problem whereas the second column presents the problem size after the transformation and shows the drastic increase

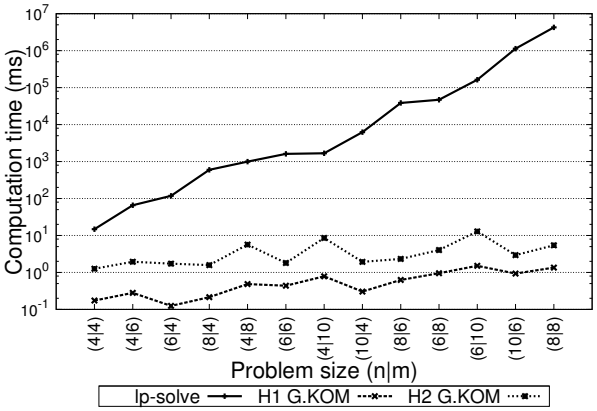

Figure 2: Problem size (n|m) vs. computation time

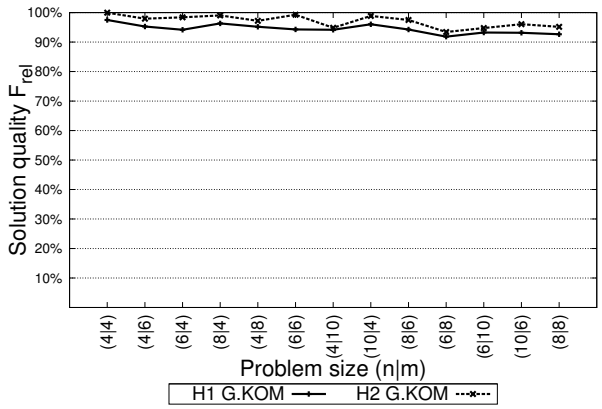

Figure 3: Problem size (n|m) vs. solution quality

in number of alternative services per category. Column 3 shows the reduced number of services per category after the adoption of the described algorithms and the reduction ratio of the search space is depicted in column 4. As a result it can be stated that dependent on the problem size the reduction of the search space has a ratio of 10^{-3} to 10^{-24}. This occurs because of the number of workflow steps and the numbers of services per category both influence the initial problem size exponentially. Assuming a problem size $n|m$ the proposed procedure of $H2\_G.KOM$ decreases the resulting aggregated services per category to less than m. Exemplarily, assuming a problem size 8|8 the power set in each category results to $2^8 = 256$ services. Thus, the new created problem is a 8|256 problem where it is only possible to select one aggregated service per category. The number of possible service combinations would lead to 256^8. This is a very large number and shows that it might not be possible to handle this problem even with a good heuristical solution. Instead, $H2\_G.KOM$ transforms the 8|8 problem into a 8|4 problem for the test cases where only one aggregated service per category has to be chosen. This yields to a very high reduction of the solution space as presented in Table 1, influenced by the workflow length and the number of services per category. Furthermore, the lower the QoS-CoS correlation, the more services can be deleted in the solution space. Because of this reduction $H2\_G.KOM$ has very good computation times at high solutions qualities as described in the following.

Evaluations show, that the developed heuristic is very fast also in comparison to $H1\_G.KOM$ as presented in Table 2, 3, and 4. As can be seen at small problem sizes, $H2\_G.KOM$ needs only 8% of the time of the exact solution. With increasing problem size the computation time of the heuristic $H2\_G.KOM$ further decreases to a relative computation time of less than 0.0001% of the exact solution. This highlights the extreme performance of $H2\_G.KOM$. Hence, comparing the relative computation time of $H2\_G.KOM$ and $H1\_G.KOM$ it can be identified that $H1\_G.KOM$ reveals always a smaller relative computation time. This observation is further presented in Figure 2 where the absolute values for the computation time of both heuristics and the exact solution are presented.

As can be seen, the exact solutions need computations times from 14 ms up to 4248259 ms depending on the problem size. Obviously, these computation times cannot be used for real-time computation of an exact solution for the depicted resource planning problem. Instead, in terms of computation time $H1\_G.KOM$ performs better than $H2\_G.KOM$ and can be used for very time critical requirements on the computation times.

Focusing on the solution quality an inverse observation can be seen, as the solution quality, depicted in Figure 3, of $H2\_G.KOM$ is always higher as in case of $H1\_G.KOM$. On average, the solution quality of $H1\_G.KOM$ has been 94.47% whereas $H2\_G.KOM$ achieves 97.11% solution quality. Furthermore the average solution quality has been never less than 91.85% for $H1\_G.KOM$ and never less than 93.42% for $H2\_G.KOM$ considering the analyzed problems.

6. CONCLUSIONS

Concerning the composition of service-based workflows invoking services with limited execution capacities, considering specific execution deadlines and workloads, the involved resource planning problem becomes very complex. As this optimization problem is proven to be NP-hard and no exact solution exists that can be solved in run-time, an efficient heuristic $H2\_G.KOM$ has been developed, implemented, and evaluated in comparison to an exact solution and another heuristical solution. The developed heuristic shows very good results concerning solution quality and computation time and is well suited for the application in time critical applications.

Our further research aims at optimizing the developed heuristical solutions and including other pricing models.

7. ACKNOWLEDGMENTS

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

8. REFERENCES

[1] R. Berbner. *Dienstgüteunterstützung für Service-orientierte Workflows*. Books on Demand, 2008.

[2] J. Eckert, D. Ertogrul, A. Miede, N. Repp, and R. Steinmetz. Resource Planning Heuristics for

Service-oriented Workflows. In *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, pages 591–597, 2008.

[3] J. Eckert, D. Ertogrul, A. Papageorgiou, N. Repp, and R. Steinmetz. The Impact of Service Pricing Models on Service Selection. In *4th International Conference on Internet and Web Applications and Services*, pages 316–321, 2009.

[4] C. Janiesch, R. Ruggaber, and Y. Sure. Eine Infrastruktur für das Internet der Dienste. *HMD – Praxis der Wirtschaftsinformatik*, 261(45):71–79, Juni 2008.

[5] W. Johannsen and M. Goeken. *Referenzmodelle für IT-Governance: Stategische Effektivität und Effizienz mit COBIT, ITIL & Co.* dpunkt.verlag, 2007.

[6] K.-J. Lin and T. Yu. Service Selection Algorithms for Composing Complex Services with Multiple QoS Constraints. In *3rd International Conference on Service-oriented Computing (ICSOC 2005)*, 2005.

[7] Z. Mahmood. Service Oriented Architecture: Potential Benefits and Challenges. In *11th WSEAS International Conference on Computers*, pages 497–501, 2007.

[8] M. P. Papazoglou. Service-Oriented Computing: Concepts, Characteristics and Directions. In *4th International Conference on Web Information Systems Engineering*, pages 3–12, December 2003.

[9] L. Zeng, B. Benatallah, A. H. Ngu, M. D. A. Kalagnanam, and H. Chang. QoS-Aware Middleware for Web Services Composition. *IEEE Transactions on Software Engineering*, 30:311–327, 2004.