MULTIOBJECTIVE MIXED-INTEGER OPTIMIZATION SCHEDULING OF WEB SERVICES

WALCIR FONTANINI[*], PAULO AUGUSTO VALENTE FERREIRA[†]

[*]*Center for Information Technology "Renato Archer"*
Rodovia Dom Pedro I (SP - 65) Km 143,6 CEP: 13069-901
Campinas - São Paulo Brazil

[†]*State Univesity of Campinas, Faculty of Electrical and*
Computer Engineering, 13082-852 Campinas, SP Brazil

Emails: `walcir.fontanini@cti.gov.br, valente@dt.fee.unicamp.br`

Abstract— The article addresses the design of an online, internet business process composed of webservices from different suppliers. We propose a design methodology based on the simultaneous optimization of several quality-of-service criteria: cost, execution time, reliability, availability and reputation. A goal programming formulation for the problem of selecting webservices suppliers results in a mixed-integer linear optimization problem. Preliminary computational tests demonstrate the validity of the approach proposed.

Keywords— E-Commerce, Scheduling, Integer Programming, Multiobjective Programming

Resumo— O artigo discute o projeto de um processo de negócios online criado a partir de serviços web disponibilizados por diferentes fornecedores. Propomos uma metodologia de projeto baseado na otimização simultanea de vários critérios de qualidade: custo, tempo de execução, confiabilidade, disponibilidade e reputação. Uma formulação do tipo Programação Alvo para o problema de seleção de provedores de serviços web resulta em um problema de otimização linear inteiro-misto. Testes computacionais preliminares demonstram a validade da abordagem proposta.

Palavras-chave— Comércio Eletrônico, Programação de Tarefas, Programação Inteira, Programação Multi-Objetivo

1 Introduction

A typical internet business can be thought as composed of several webservices to be supplied by partners with basis in internet technologies. As the number of business partners can be very large, some criteria for selecting them are necessary. We observe that most of the webservices design methodologies do not consider business performance criteria, such as cost or execution time directly, which is the focus of the present paper.

Current approaches to the internet business design problem often adopt a heuristic, trial-and-error strategy, which means that service components are assigned to individual tasks in an one-at-a-time basis. This strategy does not effectively integrate constraints of the business process and preferences of the business designer. As an example, the execution time of the composite webservice may be limited to an given value, or the total cost may not be allowed to exceed a prescribed budget.

In the present paper, a webservice design methodology based on five business performance criteria, namely, execution time, cost, service reputation, reliability and availability, is proposed. The methodology provides an optimized solution for the problem of selecting business partners (individual services). Constraints and preferences are associated with the composite service instead of with individual tasks within the composite service. As in (Zeng et al., 2004) and (Ardagna and Pernici, 2007) the services selection problem is formulated as a mathematical optimization problem.

By introducing a goal programming (multiobjective) formulation ((Gembicki and Haimes, 1975; Steuer, 1989)) for the problem, the designer can express his/her preferences about the criteria in terms of goals to be attained.

The paper is organized as follows. In section 2 a simple scenario is presented. In section 3 we detail the webservices quality model. In section 4 we formulate the webservices scheduling problem as multiobjective mixed-integer optimization problem. In section 5 we obtain and discuss some experimentals results. Conclusions are presented in section 6.

2 Example: A Travel Planner

The following representation of a business process is based on the *Business Process Management Notation* (BPMN) known as (OMG, 2008). Figure 1 presents the symbols used in the example.

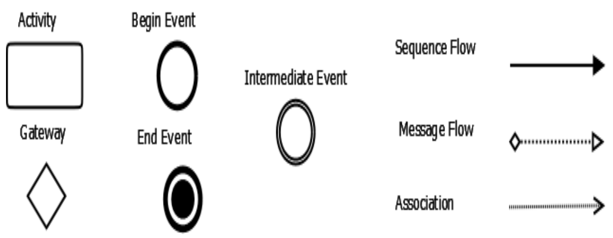

Figure 1: BPMN: Flow Objects.

Our goal here is simply to show how a reasonable scenario can be described by using BPMN.

Consider the composite webservice*s* called Travel Planner, which aggregates several composite webservice*s*, such as flight booking, travel insurance, hotel room booking, car or biking rental, and route planning. Among these, there are webservices that can be executed sequentially or concurrently.

A simplified process diagram specifying the Travel Planner composite webservice is presented in Figure 2. The user must input the destination with departure and return dates into the Travel Planner system, which then uses these informations to determine which flights are available. The next step involves choosing a hotel with available rooms. Lists with tourist sites can be provided to help in a typical travel preferences assessment. These are the most important functionalities of the Travel Planner system.

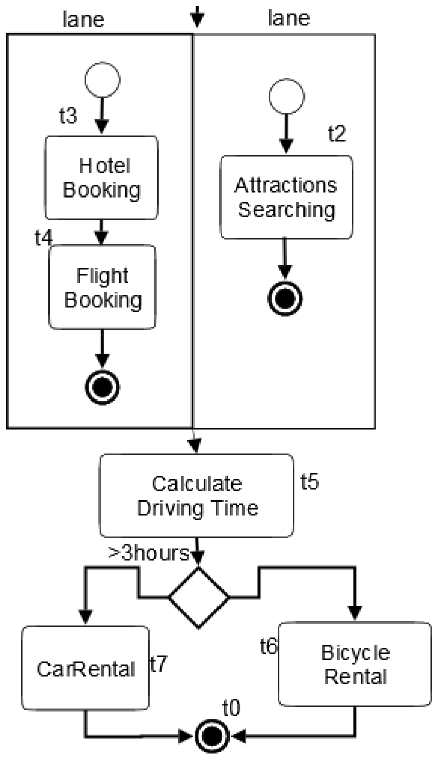

Figure 2: Travel Planner System with webservices

3 Quality Model for Webservices

A business process can be represented as a *task graph,* a terminology adopted in (Sinnen, 2007), more convenient for describing optimization problems. Since a business representation may contain IF-THEN-ELSE constructs, a possibility is to generate two separate task graphs with specific execution frequencies. An *execution path* can include parallel activities, but not activities in THEN and ELSE branches simultaneously.

The conversion rules proposed in this article are as follows. Firstly, if the original BPMN diagram contains *gateway* decisions, then a pre-processing program – a Java program that converts the BPMN diagram into a task graph – generates two graphs, one for the THEN branch and the other for the ELSE branch. Secondly, the *pool* and *lane* elements (or the horizontal parallel bars used for tasks synchronization) are not represented graphically because a depth search can find all the execution paths. Thirdly, the *activity*, *begin* and *end* elements are represented as nodes of the task graph. The graph elements are, in turn, converted into a mathematical optimization model, which considers all the quality (performance) criteria selected. Different optimization models are used in ((Ardagna and Pernici, 2005; Zeng et al., 2003)).

One concept not included in the BPMN standards, but necessary to complete the mathematical model is of *execution frequency*. When a gateway gives rise to THEN and ELSE elements, it is necessary to generate a real number between 0 and 1 associated with the relative frequency of the execution of that element. In practical implementations, the execution frequencies are obtained from recorded data.

3.1 Quality Criteria of Webservices

In (Zeng et al., 2003; Zeng et al., 2004), five quality criteria are proposed for elementary webservice*s* and then applied to composite webservices. The criteria are (1) execution cost, (2) execution time, (3) reputation, (4) reliability and (5) availability.

1. Execution Cost. For a specific webservice i from provider j, the execution cost EC_{ij} is the amount that the service client must pay for the execution of a certain operation of the webservice. Either the service providers explicitly publish their execution prices or the prices can be obtained online.

2. Execution Duration. For a specific webservice i from provider j, the execution duration ED_{ij} measures the expected delay between the execution request and the return of the output. The execution duration is calculated by the expression $ED_{ij} = T_{process} + T_{trans}$, as the execution duration is the processing time $T_{process}$ plus the transmission time T_{trans}. Webservice providers publish their processing times or provide means to obtain this information online. The transmission time is estimated from past executions of the webservice operation:

$$T_{trans} = \frac{\sum_{k=1}^{n} T_k}{n}, \qquad (1)$$

where T_k is the transmission time of the k-th observation of the transmission time, and n is the number of executions considered.

3. Reliability. The reliability R_{ij} of a webservice i from provider j is the probability that a request will be met within an expected time interval. This information should also be published online by the provider. Reliability is a measure related to software and hardware structures; it depends on the webservice server and on the quality of its connection with the service client. The reliability is calculated from recorded data by $R_{ij} = N_c/K$, where N_c is the number of correct webservice outputs delivered in the expected time interval, and K is the number of accesses considered.

4. Availability. The availability AV_{ij} of a webservice i from provider j is a probability that the service will be available. The availability is calculated by $AV_{ij} = T_a/\theta$, where T_a is the total time (in seconds) that the webservice i is available during the last θ seconds, where θ is a parameter set by the webservice administrator. The value of θ depends on the application and varies significantly. In applications in which the webservice is frequently accessed (e.g., in a stock market), small values for θ are required. If the service is less frequently used (e.g., an online bookstore), a large value for θ is more appropriate. It is possible to obtain information from the webservice providers about their availabilities.

5. Reputation. The reputation RP_{ij} of a webservice i from provider j is a subjective measure of the client satisfaction. It depends on previous experiences while using the webservice. Different users can have different opinions about the same service. Thus, reputation is defined as the average ranking of the webservice according to its users:

$$RP_{ij} = \frac{\sum_{k=1}^{n} Rank_k}{n}, \tag{2}$$

where $Rank_k$ is the ranking given by a user about the service, and n is the number of times that the webservice has been evaluated. It is a common practice that users evaluate webservices in a range from 0 to 5, for example.

3.2 Webservice Quality Aggregation

A business process diagram can have many execution paths, depending on the existence of intermediate gateways or parallel processing. In order to aggregate the quality criteria of a business process, the following notation is introduced: an *execution path* ep_l contains sequences of parallel tasks (including two special tasks, *begin* and *end*); L is the set of all execution paths; $freq_l$ is the *execution frequency* of the execution path ep_l; a *subpath* sp_m^l does not contain any parallel sequences, as it is a subset of an execution path; and an *execution plan* epl^l for an execution path ep_l is a set of ordered pairs (i, j), where the first element of the pair is the task i that belongs to ep_l and the second element is the webservice provider j.

1. Execution Duration, ED. The execution duration of a plan epl^l is the largest sum of all execution durations in each subpath sp_m^l that belongs to the execution path ep_l:

$$ED_l = \max_{sp_m^l \in ep_l} \sum_{i \in sp_m^l;(k,j)\in epl^l;k=i} ED_{ij}.$$

2. Execution Cost, EC. The execution cost of a plan epl^l for an execution path ep_l is the sum of execution costs EC_{ij} of the plan:

$$EC_l = \sum_{(i,j)\in epl^l} EC_{ij}.$$

3. Reputation, RP. The reputation of a plan epl^l for an execution path ep_l is the weighted average sum of the individual reputations RP_{ij} according to the plan epl^l; N is the number of elements in the sample:

$$RP_l = \frac{1}{N} \sum_{(i,j)\in epl^l} RP_{ij}.$$

4. Reliability, R. The reliability of a plan epl^l for an execution path ep_l is:

$$R_l = \prod_{(i,j)\in epl^l} R_{ij}.$$

5. Availability, AV. The availability of a plan epl^l for an execution path ep_l is:

$$AV_l = \prod_{(i,j)\in epl^l} AV_{ij}.$$

4 Webservices Scheduling

The webservices scheduling solution proposed in this paper is based on a policy selection that includes order parameters, agent characteristics, data on past executions and on the ongoing execution status. A budget constraint assuring that the total operational cost does not exceed a given value is also included.

4.1 A Mixed-Integer Linear Programming Solution

The optimization model below involves elements of integer programming ((Wolsey, 1998)). The variables y_{ij} indicate whether the webservice i from provider j participates in a given execution plan $(y_{ij} = 1)$ or not $(y_{ij} = 0)$.

1. **Duration and execution cost constraints.**
Let A be the set of all graph tasks. We assume that there exists a set of webservices providers WS that can be assigned to any task $i \in A$. However, for each task i, only one webservice must be assigned. The participation of the webservice providers in a given plan is subjected to the following constraint:

$$\sum_{j\in WS} y_{ij} = 1,\ \forall i \in A. \tag{3}$$

Let ST_{il} represent the earliest starting time of task i in execution path ep_l; let TD_{il} represent the execution duration of task i in execution path ep_l. The notation $j \to k$ indicates that task k directly succeeds task j. The parameter MXT is the maximum time to complete the execution of a plan. The following constraints then apply:

$$\sum_{j\in WS} ED_{ij} y_{ij} = TD_{il},\ \forall i \in A,\ \forall ep_l \in L, \tag{4}$$

$$(TD_{jl} + ST_{jl}) \le ST_{kl},\ \forall j \to k,\ \ j,k \in A, \quad \forall ep_l \in L, \tag{5}$$

$$ED_l \geqslant \sum_{i\in sp_m^l} TD_{il},\ \forall ep_l \in L, \tag{6}$$

$$ED_l \leqslant MXT,\ \ \forall ep_l \in L. \tag{7}$$

The constraint (4) imposes that the execution duration for each task i in a execution path ep_l is equal to the sum of execution durations of the webservice providers $j \in WS$ selected. Constraint (5) captures the fact that, if task k in execution path ep_l directly follows task j in the same execution path, then task k does not start before the completion of task j. Constraint (6) indicates that the execution time of each execution path is the largest among the execution times of all subpaths sp^l_m. Constraint (7) guarantees that the total execution duration of a execution path will never exceed the limit MXT. (It would be incorrect to consider the MXT as the sum of all execution paths because, actually, the run-time environment will perform only one execution path.) The parameter MXC is the amount of money available to implement any execution plan. The variable EC_l is the cost of a execution path ep_l; EC_{ij} is the cost of task i when performed by the webservice provider j. The following constraints must then hold:

$$\sum_{i \in ep_l,\, j \in WS} EC_{ij} y_{ij} = EC_l, \ \forall ep_l \in L, \tag{8}$$

$$EC_l \leqslant MXC, \ \forall ep_l \in L. \tag{9}$$

According to (8), the total cost of a execution path is the sum of the costs of the webservices selected to execute task i in the execution path ep_l. Constraint (9) indicates that the total cost of any execution path never exceeds MXC.

2. The reputation constraint.
The parameter $MNRP$ is the minimum expected reputation, while the variable RP_l is the reputation of the execution path ep_l. The parameter RP_{ij} represents the expected reputation of task i when implemented by the *w*ebservice provider j. It follows that:

$$RP_l = \sum_{i \in ep_l,\, j \in WS} RP_{ij} y_{ij} / |ep_l|, \ \forall ep_l \in L, \tag{10}$$

$$RP_l \geqslant MNRP, \ \forall ep_l \in L, \tag{11}$$

where $|ep_l|$ is the number of tasks in ep_l.

3. Availability and reliability constraints.
The parameter MNA is the minimum expected availability; the variable AV_l is the availability of the execution path ep_l. The parameter AV_{ij} represents the expected availability of the task i when implemented by the webservice provider j. The aggregation functions for availability and reliability are non-linear. Their linearizations are provided by the neperian logarithm (ln):

$$AV_l \geqslant \ln(MNA), \ \forall ep_l \in L, \tag{12}$$

$$AV_l = \sum_{i \in ep_l,\, j \in WS} \ln(AV_{ij}) y_{ij}, \ \forall ep_l \in L. \tag{13}$$

$$R_l \geqslant \ln(MNRL), \ \forall ep_l \in L, \tag{14}$$

$$R_l = \sum_{i \in ep_l,\, j \in WS} \ln(R_{ij}) y_{ij}, \ \forall ep_l \in L. \tag{15}$$

The following objectives express the selected performance criteria:

$$\min f_1 = \sum_{ep_l \in L} freq_l EC_l, \tag{16}$$

$$\min f_2 = \sum_{ep_l \in L} freq_l ED_l, \tag{17}$$

$$\max f_3 = \sum_{ep_l \in L} freq_l RP_l, \tag{18}$$

$$\max f_4 = \sum_{ep_l \in L} freq_l AV_l, \tag{19}$$

$$\max f_5 = \sum_{ep_l \in L} freq_l R_l. \tag{20}$$

where $freq_l$ is the (observed) frequency of the execution path ep_l.

4.2 *The Multiobjective Formulation of the Problem*

In this paper, the optimization problem derived from the previous analysis of webservices run-time environments is posed as a multiobjective optimization problem and solved by the well-known method *goal attainment* ((Gembicki and Haimes, 1975)). Given a set of objective functions $f_1, f_2, ..., f_m$ defined on a feasible region X, the following optimization is considered:

$$\left| \begin{array}{ll} \min \sigma & \\ s.a & f_i(x) - \sigma w_i \leq \underline{f}_i, \ i = 1, ..., m \\ & x \in X \end{array} \right. \tag{21}$$

where $\underline{f}_i$, $i = 1, .., m$, are goals for the objectives and w_i, $i = 1, .., m$, are weights associated with the goals; σ is a escalar variable. It is common to assume that $\underline{f}_i = f_{i,min}$, $i = 1, .., m$, that is, each goal is equal to the minimum of each individual objective function over the feasible region. Assuming that $w_i = \underline{f_i}$, $i = 1, .., m$, by solving (21) we minimize the largest variation of the objectives with respect to their minima. As in (16)-(20) the objectives $i = 3, 4, 5$ must be maximized, problem (21) is reformulated as follows:

$$\left| \begin{array}{ll} \min \sigma & \\ s.a & f_i(x) - \sigma w_i \leq \underline{f}_i, \ i = 1, 2 \\ & f_i(x) + \sigma w_i \geq \overline{f_i}, \ i = 3, 4, 5 \\ & x \in X \end{array} \right. \tag{22}$$

where $\overline{f_i} = f_{i,max}$, $i = 3, 4, 5$, that is, each goal is the maximum of the corresponding objective function over the feasible region X. See(Steuer, 1989) , for a review about multiobjective optimization.

5 Experimental Results

5.1 *Illustrative Example*

To numerically test our methodology, a set of random parameters – as, for example, costs – was created by means of a specific Java program that converts the execution graph into specific data structures required by the solver GnuLPK ((Gnu, 2008)). A set of execution graphs for the business process was created. These graphs are not random, as the execution graphs must be consistent. Each execution subpath that is dependent on IF-THEN-ELSE elements of the graph was executed with equal frequency. As an example, consider the business process illustrated in Figure 3 composed of three executions paths $\{ep_0, ep_1, ep_2\}$. Table 1 exhibits the parameters used for solving the associated optimization problem.

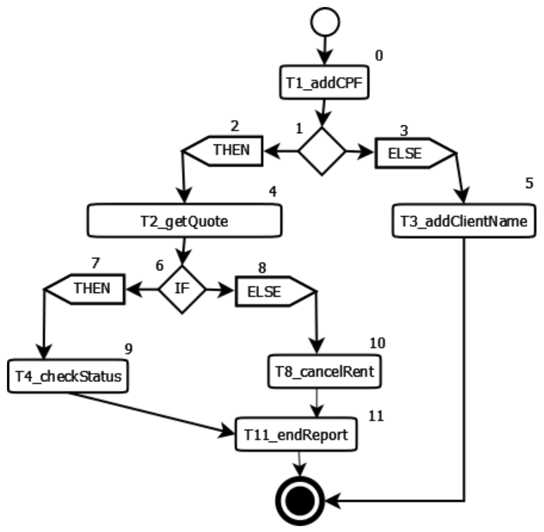

Figure 3: An example of business process in BPMN terminology.

	Expected min (or max)	ep_0	ep_1	ep_2
Cost	<=266	127	349	322
Time	<=23.74	11	29.6	30.58
Reputation	>=0.92	0.95	0.89	0.92
Availability	>=0.82	0.90	0.75	0.80
Reliability	>=0.36	0.54	0.27	0.28

Table 1: Execution parameters furnished to the solver for the example.

The time that the user waits in a queue until his/hers service starts was added to the *execution time* of the webservice.

The parameter *execution time* varied from 0.5 to 8 seconds following a uniform distribution. All tests were carried out on a micro-PC Dual-Core 3GHz with *Windows XP*.

In Table 2, a report produced by the solver is presented. The solver was initially run for each single objective in (16)-(20), generating ideal values for $cost = 266$, $time = 23.74$, $reputation = 0.9200$, $availability = 0.8169$ and $reliability = 0.3744$. The *solver* was run a sixth time to solve the goal programming problem (22) with the ideal goals obtained previously. The goal attainment method returned $cost = 356$, $time = 30.36$, $reputation = 0.8952$, $availability = 0.7941$ and $reliability = 0.3638$. Unlike the approaches proposed in (Zeng et al., 2003; Ardagna and Pernici, 2005; Ko et al., 2008) based on the minimization of a weighted sum of quality criteria, which imposes some difficulty in selecting the proper weights, the approach proposed in this paper minimizes the maximum percent variation with respect to the ideal performance values.

The sum of the execution times of all six optimization problems was less than a minute. The example had three execution paths $\{ep_0, ep_1, ep_2\}$, and the optimization report showed that no subpath violated the expected quality limits.

	Solution	**Ideal goal and weight**	**Percent variation**
Cost	356	266	0.3383
Time	30.36	23.74	0.2789
Reputation	0.8952	0.9200	0.0270
Availability	0.7941	0.8169	1.2854
Reliability	0.3638	0.3754	4.0777

Table 2: Report for 25 tasks and 10 webservice provider scenarios.

5.2 Numerical Analysis

Graphs with 5, 10 and 25 nodes (tasks) were generated using the Java program. The number of webservices providers were respectively 10, 25 and 50 . The quality parameters of the webservices were randomly varied. The parame*ter availabilit*y was generated with a uniform distribution between 0.95 and 0.99999. The parame*ter reputatio*n was similarly generated with values between 0.8 and 0.99. The parameter *reliability* was generated with a uniform distribution between 0.85 and 0.95.

Table 4 and Table 3 present the percent variations of each objective relative to their ideal values as a function of n_{ws}, the number of webservices, and m_{task}, the number of tasks.

From both Table 4 and Table 3, we observe that for a business process with five tasks, the bottleneck represented by σ^* was the execution time, even though the number of service providers had increased from 10 to 25. For businesses process with 25 tasks, the bottleneck was cost, even though the number of service providers had varied significantly: 10, 25 and 50.

Objective	$10_{ws} \times 25_{task}$	$25_{ws} \times 25_{task}$	$50_{ws} \times 25_{task}$
Cost	0.9420	1.2410	1.0697
Time	0.5459	1.1259	0.9172
Reputation	0.0528	0.0755	0.1038
Availability	0.0460	0.1220	0.1451
Reliability	0.0124	0.0286	0.3239
σ^*	0.9420	1.2410	1.0697

Table 3: Quality criteria as function of the number of webservices and tasks (Part 1).

Objective	$10_{ws} \times 5_{task}$	$25_{ws} \times 5_{task}$
Cost	1.3958	0.9479
Time	1.5692	1.3860
Reputation	0.0931	0.0638
Availability	0.0642	0.0831
Reliability	0.1479	0.0525
σ^*	1.5692	1.3860

Table 4: Quality criteria as function of the number of webservices and tasks (Part 2).

6 Conclusions

This paper presented a webservice design methodology based on multiobjective optimization in a context where internet service providers integrate webservices from several partners to deliver an expected quality-of-service to a final customer. The paper focused on the model and solution of scheduling problems associated with business processes in webservices run-time environments. The solution proposed avoids the assignment of weights to the objectives (that is, their aggregation) and, while minimizing their maximum deviation from their ideal goals, provides an useful tool for identifying the critical aspects of the implementation of a given business process in the internet.

The problem of webservices optimization has attracted considerable interest in the literature ((Ko et al., 2008; Huang et al., 2009)). The research opportunities in the field of business process optimization remain quite large. The authors currently develop some refinements of the scheduling model and its extension to interactive decision making contexts.

References

Ardagna, D. and Pernici, B. (2005). Global and local qos guarantee in web service selection, *Proceedings of Business Process Management Workshop Performance Management* pp. 36–50.

Ardagna, D. and Pernici, B. (2007). Adaptative service composition in flexible processes, *IEEE Transactions on Software Engineering* **33**(6): 369–384. DOI: 10.1109/TSE.2007.1011

Gembicki, F. and Haimes, Y. (1975). Approach to performance and sensitivity multiobjective optimization: the goal attainment method, *IEEE Transactions on Automatic Control* **AC-20**(6): 769–771. DOI: 10.1109/TAC.1975.1101105

Gnu (2008). Glpk (gnu linear programming kit), www.gnu.org/software/glpk/.

Huang, A. F., Lan, C.-W. and Yang, S. J. (2009). An optimanl qos-based web service selection scheme, *Information Sciences* **179**: 3309–3322. DOI: 10.1016/j.ins.2009.05.018

Ko, J., Kim, C. and Kwon, I.-H. (2008). Quality-of-service oriented web service composition algorithm and planning architecture, *The Journal of Systems and Software* **81**: 2079–2090. DOI: 10.1016/j.jss.2008.04.044

OMG (2008). Business process management notation from object management group, www.bpmn.org.

Sinnen, O. (2007). *Task Scheduling for Parallel Systems*, first edition edn, Wiley-Interscience, New York, USA.

Steuer, R. (1989). *Multiple Criteria Optimization: Theory, Computation and Application*, first edition edn, Krieger Pub. Co., New York, USA.

Wolsey, L. A. (1998). *Integer Programming*, first edition edn, Wiley-Interscience, New York.

Zeng, L., Benatallah, B., Dumas, M., Kalagnanam, J. and Sheng, Q. (2003). Quality-driven web services composition, *Proceedings of International World Wide Web Conference (WWW)* pp. 411–421.

Zeng, L., Benatallah, B., Ngu, A., Dumas, M., Kalagnanam, J. and Chang, H. (2004). Qos-aware middleware for web services composition, *IEEE Transactions on Software Engineer-ing* **30**(5): 311–327. DOI: 10.1109/TSE.2004.11