Spring 6-10-2017

MULTI-USER SERVICE RE-SELECTION: REACT DYNAMICALLY TO EVENTS OCCURRING AT PROCESS EXECUTION

Michael Mayer
University of Regensburg, Institute of Management Information Systems, Regensburg, Germany, michael1.mayer@ur.de

MULTI-USER SERVICE RE-SELECTION: REACT DYNAMICALLY TO EVENTS OCCURRING AT PROCESS EXECUTION

Research paper

Mayer, Michael, University of Regensburg, Institute of Management Information Systems, Regensburg, Germany, michael1.mayer@ur.de

Abstract

Considering service-based processes, the problem of determining the service candidates that fit best to a user's target weights and requirements regarding certain non-functional properties is known as QoS-aware service selection problem. Referring to multi-user processes, this requires taking into account several users with their individual goals. In this regard, users could also have preferences in the sense of user-defined requests referring to other users, so-called Inter-User-Requests (IUR). Such IUR result in dependencies among different users' service compositions that have to be taken into account when selecting services. However, due to the dynamic environment in which services are used certain events – like the failure of a service – may occur during process execution that require service re-selection at runtime. In this work, we provide such a service re-selection approach in terms of an optimization model that considers multiple users and dependencies resulting from IUR. Moreover, for the temporal coordination of the users – necessary for time-dependent IUR – we further propose a continuous time concept and integrate that in our model. Supported by our evaluation, we feel confident that this approach can serve as a first step for a comprehensive multi-user service re-selection approach where dependencies among users exist.

Keywords: Decision Support, Service Re-Selection, Multi-User Processes, Dependencies.

1 Introduction

Service-orientation as IT-architectural paradigm has received great attention in the last decade (Barry, 2012; Weinhardt et al., 2011). Encapsulating clearly defined functionalities to modular designed services enables more flexible systems (Papazoglou et al., 2007) and allows to compose single services in order to realize or support complex business processes (Sheng et al., 2014). In this regard, the continuing increase in the number of available services (e.g., currently almost 16,000 services available only on *programmableweb.com* and over 3,000 services on *appexchange.salesforce.com*) results in a growing number of functional equivalent services that differ only in their non-functional properties (NFP). These properties are represented by Quality-of-Service (QoS) criteria such as costs, duration or availability (Alrifai et al., 2012). This development might be additionally reinforced by the recent rise of the microservice architectural style which postulates to design applications as independently deployable services (Lewis and Fowler, 2014). As a consequence, there exists a decision problem that is related to the question which services fit best to each single action (i.e. service class) of the underlying process for a certain user (Alrifai et al., 2012). For this, the user's individual target weights and requirements (e.g., constraints with respect to the end-to-end process) regarding the NFP can be taken into account (Zeng et al., 2004).

The problem of determining the optimal service composition is known as QoS-aware service selection problem and has been widely discussed in literature for single user processes (e.g., Alrifai and Risse, 2009; Ardagna and Mirandola, 2010; Yu et al., 2007; Zeng et al., 2004) as well as for multi-user processes (e.g., Benouaret et al., 2012; He et al., 2012; Heinrich et al., 2015a; Kang et al., 2011; Wang et al., 2010). Multi-user processes require to deal with several users and their individual target weights and

requirements. Furthermore, there may also exist dependencies among the service compositions of different users that have to be taken into account in multi-user service selection. In this respect, existing multi-user approaches consider dependencies resulting from hard restrictions (such as predetermined capacity limits) on the one side (e.g., Benouaret et al., 2012; He et al., 2012; Kang et al., 2011; Wang et al., 2010) and user preferences on the other side. Such user preferences could be user-defined requests referring to other users, so-called Inter-User-Requests (IUR) (Heinrich et al., 2015a), for instance, a user wants to use (or not) a specific service together with certain other users.

However, due to the dynamic environment in which services are used (cf. e.g., Sheng et al., 2014), services selected at planning time may, for instance, take longer than expected, become unavailable or even fail during their execution (cf. Canfora et al., 2008; Sheng et al., 2014; Zheng et al., 2014). Therefore, there may exist new optimal service compositions at runtime or the initially planned service compositions may even be infeasible. For instance in case of a service failure, there may possibly exist numerous alternative service candidates a user could select as substitute. The user would then also need to consider the already executed part of the process as well as the remaining part to ensure the new service composition is still feasible (e.g., due to the user's NFP constraints). As a result, the user might see her-/himself being confronted with an information overload problem (cf. Shen et al., 2012a) – where decision support could be provided by a suitable service re-selection approach. Although there exist some service selection approaches which consider the effects of potential service failures already at planning time (e.g., Heinrich et al., 2015b; Yu and Lin, 2005), a dynamic re-selection approach that reacts to service failures and other events at runtime is still necessary. Additionally, in the context of multi-user processes, events occurring for one user may also influence other users' process execution due to existing dependencies among the users. However, existing optimization-based service re-selection approaches consider only single user processes and therefore no such dependencies (e.g., Berbner et al., 2007; Canfora et al., 2008; Li et al., 2011; Sandionigi et al., 2013).

Thus, the aim of this work is to propose a novel multi-user service re-selection approach in terms of an optimization model that allows to react dynamically to events occurring at process execution under consideration of dependencies among different users' service compositions (contribution ❶). Here, we focus on dependencies resulting from user preferences in the sense of IUR. Moreover, we distinguish mutual (time-independent) and simultaneous (time-dependent) IUR. For the consideration of the latter a temporal coordination of the users' actions is required. For this purpose, we need to develop a concept dealing with time as continuous (contribution ❷) since in service re-selection at runtime events could occur at any time. By this, our work refers to the following research questions:

How to design a dynamic optimization-based multi-user service re-selection approach that is capable of considering the effects of events occurring at process execution? How to integrate a continuous time concept within this multi-user service re-selection?

The remainder of this paper is structured as follows: In the next section, we analyze and discuss the existing literature related to the identified research gap and our contribution. This is followed by the introduction of our model setup including the definition of IUR. Based on that, we develop the continuous time concept and integrate this in our novel multi-user service re-selection approach in Section 5. In Section 6, we then provide an evaluation of this approach. We conclude our work with a short discussion on limitations and future research.

2 Related Literature, Research Gap and Contribution

Since our research is related to the consideration of dependencies among different users in multi-user service selection problems, we first analyze the existing approaches in that field, before we discuss the identified research gap and our contribution with respect to literature on QoS-aware service re-selection.

As already described, with multiple users participating in a process the common single user service selection problem is extended by the consideration of potential dependencies among different users' service compositions. Existing works could be divided in approaches focussing on hard restrictions and

approaches considering user preferences when determining the (near) optimal service compositions for all users under consideration of their individual target weights and requirements regarding the NFP. In terms of hard restrictions, which have to be satisfied in a feasible service composition, this particularly refers to the consideration of capacity limits of services (e.g., He et al., 2012; Jin et al., 2012; Kang et al., 2011; Shen et al., 2012b) as well as the mandatory mutual use of a certain service by several users (e.g., Benouaret et al., 2012; Wanchun et al., 2011; Wang et al., 2010). On the other side, user preferences obviously affect the utility and thus the optimality of a service composition. Here, to the best of our knowledge only Heinrich et al. (2015a) provide an approach that enables to consider user preferences, so-called user-defined requests referring to other users (IUR). The multi-user service selection approach proposed by them utilizes an optimization model formulated as knapsack problem for selection of the optimal service compositions for all users *at planning time* while taking into account the dependencies resulting from IUR. Furthermore, for consideration of simultaneous IUR they suggest a concept dealing with *time as discrete* by introducing special waiting service classes and waiting services.

In this work, we focus on multi-user service re-selection at runtime, which means the consideration of events occurring during execution of the initially planned service compositions. Re-selection may be required or appropriate, for instance, after the failure of services or the appearance of new services (cf. Ardagna and Pernici, 2007; Canfora et al., 2008), the deviation of realized from expected NFP values (cf. Canfora et al., 2008; Shen et al., 2012b), or users redefining their target weights (cf. Ardagna and Pernici, 2007) and requirements (cf. Shen et al., 2012b). Referring to multi-user processes, such events could also be users leaving the process or the participation of new users. Besides approaches that consider potential service failures already at planning time (e.g., Heinrich et al., 2015b; Yu and Lin, 2005) or following a certain fault-tolerant strategy (e.g., Shen et al., 2012b; Stein et al., 2009; Zheng and Lyu, 2010), several optimization-based service re-selection approaches have been proposed to deal with unforeseen events occurring at runtime. In this respect, they aim at enabling successful process completion by determining a new feasible, (near) optimal service composition for a single user. While Ardagna and Pernici (2007), Sandionigi et al. (2013) and Zeng et al. (2004) simply suggest to apply their proposed optimization model regarding service selection at planning time on the remaining part of the process in case of an event at runtime, Berbner et al. (2007), Canfora et al. (2008), Li et al. (2011) and Lin et al. (2010) provide independent service re-selection approaches. Li et al. (2011) and Lin et al. (2010), for instance, propose an iterative approach which gradually expands the part of the process considered in re-selection and thus trying to find a solution that does not violate the constraints regarding the NFP. On the other side, both Berbner et al. (2007) and Canfora et al. (2008) provide a heuristic approach applied on the whole remaining process that determines the new near optimal service composition for the user. Regarding this, the algorithm H1_RELAX_IP of Berbner et al. (2007) uses the LP relaxation of the original mixed integer problem combined with a backtracking algorithm, while Canfora et al. (2008) developed an approach based on a genetic algorithm. Besides that, in the field of semantic web services, existing approaches (e.g., Klusch and Kapahnke, 2012; Li et al., 2008; Rodriguez-Mier et al., 2012) could possibly support the QoS-aware service re-selection by automatically discovering new functionally equivalent services, for instance, in case the currently executed service fails.

Research Gap and Contribution to Research

According to Ardagna and Pernici (2007), a valid re-selection to determine the new optimal service composition after occurrence of a (runtime) event in single user problems seems to be to simply apply a service selection approach again on the remaining part of the process. However, using this idea especially in the context of multi-user processes and IUR would not necessarily lead to a feasible and optimal solution: On the one hand, the impact of the occurred event itself could not be considered correctly which may lead to infeasible service compositions (e.g., realized NFP of a failed service affect users' constraints). On the other hand, dependencies existing between different users' services located in the already executed and the remaining part of the process would be disregarded. Thus, when considering IUR and the resulting dependencies, the entire initial process has to be taken into account. Furthermore, events – although directly related only to one user – might also affect other users' service compositions (e.g., a user leaving the process). In this respect, there could be IUR that are planned to be realized in

the initial service compositions but will not be realized due to unforeseen events, or the other way round. We therefore aim at providing a service re-selection approach that – after occurrence of an event at runtime – considers for all users the already executed (or currently executing) services and realized NFP as well as the still unexecuted actions of the remaining part of the process (contribution ❶).

Furthermore, using a discrete time concept – as proposed in (Heinrich et al., 2015a) – for the temporal coordination of the users' actions might be sufficient in many situations, but in service re-selection at runtime this would be accompanied by some serious weaknesses regarding flexibility and performance (see Section 4.1). Because of this, we propose a concept that enables to consider time as continuous (contribution ❷). As to the best of our knowledge, there exists no work within service science describing an optimization model that would allow to consider multiple users with dependencies among them where the model also contains a continuous time concept for temporal coordination of the users, we develop such an optimization model in this work.

3 Model Setup

In the following, we introduce our model setup in terms of those definitions and modeling elements that can serve as common knowledge base. This allows for a better differentiation between existing knowledge and our contribution later on.

In this work, we consider a process with multiple participating users $a \in A$. More precisely, a process consists of a number of actions or service classes i (with $i = 1$ to I), respectively, that contribute to achieve an intended goal. Each service class encompasses a set of functional equivalent service candidates s_{ij} that only differ in their non-functional properties (NFP) represented by Quality-of-Service (QoS) attributes (e.g., costs, duration, availability). Furthermore, a service composition is defined as a concrete implementation of a process in terms of a set of services with exactly one service candidate out of each service class of the process.

3.1 NFP and Utility Function

We further define NFP as the set of attributes N that have to be considered in service selection or re-selection, respectively. This set N can be divided into the subset of attributes N^- that need to be minimized, the subset of attributes N^+ that need to be maximized, and the subset of attributes N^{tv} that refer to a target value tv. Further, we introduce the vector $q_{ij} = \left[q_{ij}^1, \dots, q_{ij}^N\right]^T$ of the quantified NFP values of a service candidate s_{ij}. When selecting service candidates with several NFP, we use – in line with the existing literature – a utility function which aggregates the values of the different attributes N to a single utility value U (cf. e.g., Ai and Tang, 2008; Alrifai and Risse, 2009; Ardagna and Pernici, 2007; Yu et al., 2007). In our work, we apply the utility function described, e.g., by Alrifai and Risse (2009), which is based upon the simple additive weighting (SAW) technique: First, the values $q_{ij}^{\propto}$ of all attributes $\propto \in N$ are normalized in the interval [0;1] to enable comparability between different attributes. For this, the aggregated maximum $P_{max}^{\propto}$ and minimum $P_{min}^{\propto}$ of the attributes N^- and N^+ – and $P_{max^*}^{\propto}$, $P_{min^*}^{\propto}$ for N^{tv} – over all service classes S_i are used, which can be calculated as follows:

$$P_{max}^{\propto} = \sum_{i=1}^{I} P_{i,max}^{\propto} = \sum_{i=1}^{I} \max_{s_{ij} \in S_i} q_{ij}^{\propto}; \quad P_{min}^{\propto} = \sum_{i=1}^{I} P_{i,min}^{\propto} = \sum_{i=1}^{I} \min_{s_{ij} \in S_i} q_{ij}^{\propto} \tag{1}$$

$$P_{max^*}^{\propto} = \sum_{i=1}^{I} P_{i,max^*}^{\propto} = \sum_{i=1}^{I} \max_{s_{ij} \in S_i} \left(\left|q_{ij}^{\propto} - tv^{\propto}\right|\right); \quad P_{min^*}^{\propto} = \sum_{i=1}^{I} P_{i,min^*}^{\propto} = \sum_{i=1}^{I} \min_{s_{ij} \in S_i} \left(\left|q_{ij}^{\propto} - tv^{\propto}\right|\right) \tag{2}$$

The utility score of a single service candidate could then be determined by taking the weighted sum over all attributes under consideration of user-defined target weights regarding the attributes N. With multiple participating users each user $a \in A$ may possibly have its individual target weights $w_a^{\propto}$ (with $\sum_{\propto=1}^{N} w_a^{\propto} = 1$), leading to varying utility scores $U_{a_{ij}}$ for the same service candidate s_{ij} but different user a.

$$U_{a_{ij}} = \sum_{\propto \in N^-} \left(\frac{P^{\propto}_{i,max} - q^{\propto}_{ij}}{P^{\propto}_{max} - P^{\propto}_{min}} \right) * w^{\propto}_a + \sum_{\propto \in N^+} \left(\frac{q^{\propto}_{ij} - P^{\propto}_{i,min}}{P^{\propto}_{max} - P^{\propto}_{min}} \right) * w^{\propto}_a + \sum_{\propto \in N^{tv}} \left(\frac{P^{\propto}_{i,max^*} - (|q^{\propto}_{ij} - tv|)}{P^{\propto}_{max^*} - P^{\propto}_{min^*}} \right) * w^{\propto}_a \quad (3)$$

By summing up the utility scores of all selected services by all users the overall utility value of a service composition could be determined. In line with existing optimization-based approaches, we formulate our optimization model provided in Section 4 as knapsack problem (e.g., Alrifai et al., 2012; Strunk, 2010; Yu et al., 2007). Thus, it consists of an objective function determining the overall utility value and several constraints, for instance, to integrate the users' global end-to-end requirements regarding the NFP, which can be described by the vector $Q_a = [Q_a^1, \dots, Q_a^N]^T$. In this respect, we use decision variables $x_{a_{ij}}$ for each user $a \in A$ and every service candidate s_{ij}, with $x_{a_{ij}} = 1$ indicating that service candidate s_{ij} is selected for user a, and $x_{a_{ij}} = 0$ that is not.

3.2 Considering Inter-User-Requests

According to (Heinrich et al., 2015a), IUR are specified as user-defined requests referring to other users. In contrast to hard restrictions as considered, e.g., in (Benouaret et al., 2012; He et al., 2012; Kang et al., 2011; Wang et al., 2010), IUR represent user preferences assessing different alternatives, for example, using a certain service together with defined other users or not. Generally, an IUR is defined by a user who determines the service or service class related to that IUR and the set of participating users. In scenarios where a user does not know all other users in the process, the user could instead describe the participating users of an IUR by certain characteristics such as age, gender or interests in terms of persons as users and industry branch, country or company size in terms of companies. Based on the described characteristics, the corresponding group of users could be identified and connected to that IUR. Furthermore, a user associates a certain (positive or negative) value with the realization of an IUR. In line with Heinrich et al. (2015a), we distinguish four basic types of IUR, regarding the two dimensions relation and time:

	Complementary	**Conflicting**
Mutual (time-independent)	*Complementary mutual usage:* • A user requests to perform an action **together** with one or more other users. • A **positive value** is associated with this IUR.	*Conflicting mutual usage:* • A user requests **not** to perform an action together with one or more other users. • A **negative value** is associated with this IUR.
Simultaneous (time-dependent)	*Complementary simultaneous usage:* • A user requests to perform and thus to start an action **together** with one or more users **at the same time**. • Potential occurrence of **waiting times** for users. • A **positive value** is associated with this IUR.	*Conflicting simultaneous usage:* • A user requests **not** to perform an action together with one or more other users at **any given moment of time**. • Potential occurrence of **waiting times** for users. • A **negative value** is associated with this IUR.

Table 1. Fundamental types of IUR (Heinrich et al., 2015a)

Based on that, each user may specify a set of IUR E_a^{IUR}, where a single IUR $e \in E_a^{IUR}$ could be formally defined by the following quadruplet:

$$e = (\hat{U}_e, \bar{\bar{U}}_e, A_e, X_e) \quad (4)$$

An IUR e, thereby, is defined by means of the utility $\widehat{U}_e$ (which is distinct from 0 in case e is a mutual IUR), the utility $\overline{\overline{U}}_e$ (which is distinct from 0 in case e is a simultaneous IUR), the set of participating users A_e and the set X_e of corresponding decision variables $x_{a_{ij}}$.

Furthermore, for the consideration of simultaneous (i.e. time-dependent) IUR a temporal coordination of the users' actions is needed. In this respect, it may possibly be more beneficial for a user to wait a certain amount of time, for instance, to realize the positive utility associated with a complementary simultaneous IUR or to avoid the negative utility associated with a conflicting simultaneous IUR. This requires a concept to consider potential waiting times as well as the loss of utility caused by waiting in the corresponding optimization model. More precisely, there has to be the possibility for a user to wait right between two succeeding actions. For this, Heinrich et al. (2015a) propose a concept which considers time (in terms of duration, response time, etc.) and waiting time of a service selection problem as discrete. Particularly, they introduce waiting time WT as additional NFP and special waiting service classes S_i^* right in front of each regular service class S_i, with each waiting service class encompassing a set of waiting services s_{ij}^*. Attributes representing "time" (i.e. duration/ response time Dur, waiting time WT) are modeled as discrete values $q_{ij}^{Dur}, q_{ij}^{WT} \in \{k * c | \, c \in \mathbb{R}^+\}$ with $k \in \{0, 1, \dots, K\}$ and thus in discrete steps: Every waiting service class consists of a defined number of waiting services, each being described by a different specific waiting time q_{ij}^{WT} and thus related to a different utility score as the utility is calculated similar to regular service candidates. However, values q_{ij}^{Dur} representing NFP duration, response time, etc. must also fit to these discrete time steps. The parameter c specifies the fixed length of each time interval. For example, making c smaller results in more discrete steps necessary to cover the same overall time range in the corresponding optimization model. Thereby, the optimization model evaluates the different alternatives (e.g., realization of $\overline{\overline{U}}_e$ vs. loss of utility through waiting) and determines the right waiting service candidate, that means the right (discrete) amount of waiting, for each waiting service class and user of the process.

Moreover, as IUR affect more than one user, the consideration of IUR results in dependencies between different users' service compositions. In the case of simultaneous IUR, these dependencies are also of temporal nature. Regarding the formulation of a service selection problem as optimization model, the dependencies resulting from mutual IUR can be integrated directly in the objective function (of a non-linear model), whereas temporal-based dependencies related to simultaneous IUR require additional constraints (cf. Heinrich et al., 2015a). By solving such an optimization model, the initial optimal service compositions for all users regarding a certain multi-user service selection problem can be determined.

4 Novel Multi-User Service Re-Selection Approach

In this section, we present our service re-selection approach which enables to consider multiple users with their IUR when re-optimizing the users' initial service compositions after occurrence of a certain event during process execution. Subject to the event, there can be distinguished three general complementary goals for performing service re-selection (cf. Berbner et al., 2007):

(1) *Recovery*: To enable successful process completion for a user (e.g., after failure of a service).

(2) *Feasibility*: To ensure the selected service composition for each user is feasible (e.g., in case the realized NFP differ significantly from the ex-ante expected values).

(3) *Optimality*: To ensure the optimal service composition is selected for each user (e.g., after failure of a service or a user leaving the process).

Thus, the overall objective is to make certain that after occurrence of an event the process execution is still possible and feasible for all users – under the consideration that possibly new optimal service compositions for the individual users may exist. In order to take all global end-to-end constraints of the users and potential dependencies resulting from mutual and simultaneous IUR into account, it is necessary to adopt a global perspective, that means to consider the entire process, for re-optimization of the remaining part.

In the next paragraph, we therefore propose a continuous time concept enabling the consideration of simultaneous IUR and the resulting temporal-based dependencies in re-selection (contribution ❷). Based on that, we then describe how to integrate multiple users, IUR and this concept in an optimization model for service re-optimization of the remaining part of the process and all users (contribution ❶). For a discussion on the technical aspects regarding detection and triggering service re-selection we refer, e.g., to Canfora et al. (2008), Lin et al. (2010) or Shen et al. (2012b).

4.1 Concept for Continuous Consideration of Time

For the temporal coordination of the users' actions required for consideration of simultaneous IUR we theoretically could adopt the existing discrete time concept described in Section 3.2. In this case, for each service selection problem a suitable length for the discrete time intervals (i.e. parameter c) would have to be defined when building the optimization model. In terms of service selection at planning time rather large time intervals seem to be sufficient in most cases (cf. Heinrich et al., 2015a). But the premises change when considering service re-selection at runtime as the execution of services or waiting could be disrupted at any given moment, for example, a service could fail at any time. Furthermore, the actual realized execution time for a service and actual waiting times do generally differ from the discrete values used for service selection at planning time. Thus, applying the described discrete time concept in a re-selection approach would require to specify much smaller intervals compared to service selection at planning time. However, this would result in problems regarding

- *Flexibility*: What is the optimal choice for the length of a time interval (i.e. parameter c)?
- *Performance*: Smaller time intervals normally correspond to larger problem sizes and thus higher computation times for calculating the optimal solution.

In the following, we therefore propose a concept to integrate (waiting) time as continuous in an optimization model: Usually, the utility of a user's service composition is determined by adding up the a-priori calculated utilities of the selected service candidates in the objective function of the optimization model (cf. Section 3.1). Using the described utility function the utility of a user's service composition can also be calculated during solving of the optimization model based on the aggregated NFP values of the service composition as the following equation illustrates:

$$\sum_{a \in A} \sum_{i=1}^{I} \sum_{s_{ij} \in S_i} U_{a_{ij}} x_{a_{ij}} = \sum_{a \in A} \sum_{\propto \in N} w_a^{\propto} \frac{P_{max}^{\propto} - \left(\sum_{i=1}^{I} \sum_{s_{ij} \in S_i} q_{ij}^{\propto} x_{a_{ij}} \right)}{P_{max}^{\propto} - P_{min}^{\propto}} \tag{5}$$

In contrast to calculate the utility of a service candidate a-priori based on predetermined NFP values, the term on the right allows to use a variable instead of a fixed value $q_{ij}^{\propto}$ for an attribute $\propto$ where the optimal value is then determined by the optimization model. Because of that, waiting time can be modeled as continuous by using variables $wt_{a_i} \in \mathbb{R}_0^+$ for NFP waiting time and let the optimization model dynamically determine the right amount of waiting time wt_{a_i} and the corresponding utility during solving of the model. Particularly, we connect a waiting time variable wt_{a_i} with each service class i and each user a (cf. Figure 1).

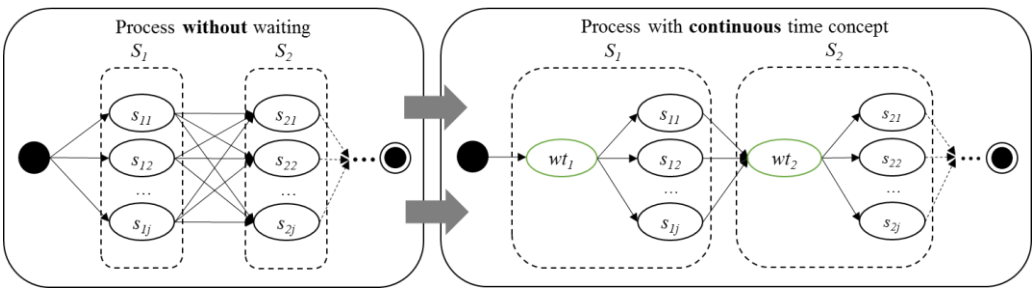

Figure 1. Illustration of process with continuous time concept for a single user

By this, there is no need to prescribe fixed, discrete values for waiting when building the optimization model. Consequently, there is also no restraint on certain discrete values for attributes representing duration/ response time, and thus $q_{ij}^{Dur} \in \mathbb{R}^{+}$. Therefore, the continuous concept overcomes the problem regarding *flexibility* related to the discrete time concept. Furthermore, it generally also *outperforms* the existing concept in terms of computation time required for calculating the optimal solution (cf. Section 5.1).

4.2 Optimization Model for Service Re-Selection

Hereafter, we introduce our optimization model for multi-user service re-selection. In case the re-selection is triggered (after occurrence of an event), the process can be divided into three regions for each user (cf. Zeng et al., 2004): Region (A) of already completely executed services and waited times, region (B) of currently being executed services or waiting, and region (C) of still unexecuted but planned services and waiting times. To be able to consider dependencies resulting from IUR that may exist among users' services/ service classes in different regions, we formulate the corresponding optimization model for the entire process taking into account regions (A)-(C).

Moreover, as a service could fail during its execution, we also have to take a possible "consume" of NFP (e.g., time) into consideration. Focusing, for instance, on attributes representing time this means that although the failed service could not be executed completely, the amount of time until detection of the failure is nevertheless consumed. As this affects the global end-to-end NFP constraints and also the temporal coordination of users' unexecuted actions, we therefore add to our model the variables $CA_{a_i}^{\propto}$ for each user a and each service class i that holds the consumed amount for each attribute $\propto \in N \backslash \{WT\}$[1].

In the following, we describe our non-linear optimization model for the consideration of multiple users and IUR in service re-selection which also integrates the proposed concepts for continuous time and consumed NFP. It is formulated as knapsack problem, consisting of an objective function and several constraints: The objective function of our model determines the accumulated maximum utility over all users $a \in A$, all service classes S_i and all service candidates s_{ij} by taking into account the binary decision variables $x_{a_{ij}}$ and s_e as well as the continuous waiting time variables wt_{a_i}:

$$\max_{x_{a_{ij}}; wt_{a_i}; s_e} \sum_{a \in A} \sum_{\substack{\propto \in N \\ \backslash \{WT\}}} w_a^{\propto} \frac{P_{max}^{\propto} - \left(\sum_{i=1}^{I} \left(\sum_{s_{ij} \in S_i} q_{ij}^{\propto} x_{a_{ij}} + CA_{a_i}^{\propto}\right)\right)}{P_{max}^{\propto} - P_{min}^{\propto}} + \sum_{a \in A} w_a^{WT} \frac{P_{a,max}^{WT} - \sum_{i=1}^{I} wt_{a_i}}{P_{a,max}^{WT} - P_{a,min}^{WT}} + \sum_{a \in A} \sum_{e \in E_a^{IUR}} \widehat{U}_e \prod_{x_{a_{ij}} \in X_e} x_{a_{ij}} + \sum_{a \in A} \sum_{e \in E_a^{IUR}} \overline{\overline{U}}_e s_e \prod_{x_{a_{ij}} \in X_e} x_{a_{ij}} \tag{6}$$

More precisely, the first summand calculates the utility of the users' service compositions based on the aggregated NFP values – including possibly consumed NFP $CA_{a_i}^{\propto}$, but without waiting time WT – of the selected service candidates ($x_{a_{ij}} = 1$ indicates that service candidate s_{ij} is selected for user a, $x_{a_{ij}} = 0$ that is not). The second summand determines the utility subject to the amount of waiting time WT for all users, the third and fourth summand refer to determining the utility regarding IUR. Moreover, the utility $\widehat{U}_e$ of a mutual IUR $e \in E_a^{IUR}$ is realized if – and only if – all decision variables $x_{a_{ij}} \in X_e$ are 1, that means, all corresponding service candidates s_{ij} have to be part of the solution. For the realization of $\overline{\overline{U}}_e$ of a simultaneous IUR in addition the indicator variable s_e has to be 1. This variable s_e is linked to the following constraints which evaluate whether the temporal conditions associated with a complementary (7) or conflicting (8) simultaneous IUR are fulfilled or not:

[1] Whether there has to be considered consumed NFP of an attribute $\propto \in N$ depends on the specific type of attribute or, for instance, on the contractual agreements (e.g., SLA) of user and service provider.

$$\left\lfloor \max_{\left\{\substack{a\in A_e| \\ x_{a_{i'j'}}\in X_e}\right\}} \left(\sum_{i=1}^{i'-1} \left(wt_{a_i} + \sum_{s_{ij}\in S_i} q_{ij}^{Dur} x_{a_{ij}} + CA_{a_i}^{Dur} \right) + wt_{a_{i'}} \right) - \min_{\left\{\substack{a\in A_e| \\ x_{a_{i'j'}}\in X_e}\right\}} \left(\sum_{i=1}^{i'-1} \left(wt_{a_i} + \sum_{s_{ij}\in S_i} q_{ij}^{Dur} x_{a_{ij}} + CA_{a_i}^{Dur} \right) + wt_{a_{i'}} \right) \right\rfloor * s_e \leq 0 \quad \forall\, e \in E_a^{IUR}, \overline{\overline{U}}_e > 0 \tag{7}$$

$$\left\lfloor \max_{\left\{\substack{a\in A_e| \\ x_{a_{i'j'}}\in X_e}\right\}} \left(\sum_{i=1}^{i'-1} \left(wt_{a_i} + \sum_{s_{ij}\in S_i} q_{ij}^{Dur} x_{a_{ij}} + CA_{a_i}^{Dur} \right) + wt_{a_{i'}} \right) - \min_{\left\{\substack{a\in A_e| \\ x_{a_{i'j'}}\in X_e}\right\}} \left(\sum_{i=1}^{i'-1} \left(wt_{a_i} + \sum_{s_{ij}\in S_i} q_{ij}^{Dur} x_{a_{ij}} + CA_{a_i}^{Dur} \right) + wt_{a_{i'}} + q_{i'j'}^{Dur} x_{a_{i'j'}} \right) \right\rfloor * (1 - s_e) \geq 0 \quad \forall\, e \in E_a^{IUR}, \overline{\overline{U}}_e < 0 \tag{8}$$

In terms of constraints (7), which refer to complementary simultaneous IUR, s_e is 1 if the service compositions of the users $a \in A_e$ all possess the same duration until the point in time right before the potential invocation of the considered service candidates in X_e. Regarding constraints (8) and conflicting simultaneous IUR, s_e could get 0 – to avoid the associated negative utility – if there exists no point in time where the execution of all service candidates $x_{a_{i'j'}} \in X_e$ is overlapping. For the calculation of the duration of a user's service composition until a certain service class $S_{i'}$ both terms (7) and (8) take into account the duration/ response time q_{ij}^{Dur} of the selected services, the waiting time wt_{a_i} and possibly consumed time $CA_{a_i}^{Dur}$. By adjusting the users' decision variables for the individual service candidates and waiting time variables the optimization model enables the temporal coordination of the users' actions in order to achieve the overall optimal solution.

$$\sum_{i=1}^{I} \left(\sum_{s_{ij}\in S_i} q_{ij}^{\propto} x_{a_{ij}} + CA_{a_i}^{\propto} \right) \leq Q_a^{\propto} \quad \forall \propto \in N \backslash \{WT\};\ \forall\, a \in A \tag{9}$$

$$\sum_{i=1}^{I} wt_{a_i} \leq Q_a^{WT} \quad \forall\, a \in A \tag{10}$$

Moreover, the users' global end-to-end constraints regarding the NFP are taken into account by means of term (9) – for all attributes $\propto \in N \backslash \{WT\}$ and under consideration of possibly consumed NFP $CA_{a_i}^{\propto}$ – and term (10) for waiting time WT.

Finally, constraints (11) make certain that exactly one service candidate s_{ij} is selected for each service class S_i and each user a:

$$\sum_{s_{ij}\in S_i} x_{a_{ij}} = 1 \quad \forall\, i = 1, \dots, I;\ \forall\, a \in A \tag{11}$$

However, to ensure correct service re-optimization, our hitherto described basic optimization model has to be adjusted subject to the specific characteristics of the event causing the re-selection – which concerns the following elements:

- The impact of the event itself (failed service, left user, diverging NFP values, etc.)
- Region (A): The already completely executed services and waited times of each user
- Region (B): The current state of each user

We integrate these into our model by modifying existing and adding additional constraints. First, the consideration of the event itself highly depends on the type of the event. For instance, a failure of service

s_{ij} for user a can be taken into account by setting the corresponding decision variable $x_{a_{ij}} = 0$. If the failure occurs during execution of the service and thus NFP are consumed, in addition the values $CA_{a_i}^{\propto}$ have to be set accordingly. This is also true in case a user a leaves the process during execution of a service. Even though the user leaves, we do not completely extract her/him from the optimization model as there may still exist dependencies to other users' service compositions, e.g., between region (A) of the leaving user and region (C) of other users. Indeed, to model that a user leaves right before or during service class i' and thus is not participating in the process any further, we set constraint (11) and all waiting time variables wt_{a_i} to zero for all service classes $i = i'$ to I. In terms of a service candidate's NFP values differing from the expected values, the model is adjusted by updating the affected $q_{ij}^{\propto}$ regarding that service. Considering region (A), already completely executed services and waited times are integrated by setting the corresponding decision variables $x_{a_{ij}} = 1$ and waiting time variables $wt_{a_i} = V$, with V as the actual waited time. Besides that, the ex-ante expected NFP values $q_{ij}^{\propto}$ can be replaced by the actual realized ones. With regard to region (B), if a user is currently executing a service, the related decision variable $x_{a_{ij}}$ is set to 1. In case a user is currently waiting, s/he could either continue or stop waiting. This is considered by integrating the corresponding waiting time variable $wt_{a_i} \geq V$ with V as the already waited time. For region (C) – that means the unexecuted actions of the process – no further adjustments to the basic optimization model are necessary.

Based on this optimization model tailored to the associated event, the optimal solution for the service re-selection problem in hand could be determined, for instance, by applying mixed integer programming (cf. Nemhauser and Wolsey, 1988). Further, if a new optimal service composition is found for one or more users and the remaining part of the process, it may then be deployed and executed.

5 Evaluation

In this section, we evaluate our approach. We first compare our continuous time concept with the existing discrete concept in terms of computation time. By this, we want to analyze whether our concept can overcome the performance issues that would occur when using the existing concept in service re-selection. Second, we want to demonstrate the efficacy of our multi-user service re-selection approach based on a real-world scenario. In this regard, our evaluation design follows the compositional styles simulation- and metric-based benchmarking of artefacts and demonstration (cf. Prat et al., 2015).

To enable the application of mixed integer programming in order to obtain the optimal solution for our optimization model, we first had to linearize our presented non-linear model (using the guidelines as proposed by, e.g., Williams, 2013). Moreover, we implemented this linearized version in Java, and – to ensure a correct implementation – we further conducted intensive testing of the source code (i.e. manual analysis by other persons than the programmers, unit tests, regression tests, runs with extreme values). For solving the model we use the mathematical programming solver Gurobi[2].

5.1 Performance

As described in Section 4.1, using the existing discrete time concept in multi-user service re-selection would result in performance issues since this requires small time intervals. In the following, we want to analyze these performance issues and whether our novel continuous time concept can overcome them.

For this purpose, we evaluate the computation time needed for solving an exemplary multi-user service selection problem with our approach (this equals re-selection without an event and thus our basic optimization model) and an approach using the discrete time concept, and compare the results. More precisely, for the discrete time approach we consider several settings, each increasing the number of regarded time intervals by reducing the parameter c (i.e. the fixed length of each time interval) while

[2] http://www.gurobi.com/, accessed August 2016

keeping all other parameters unchanged (ceteris paribus). Our representative problem (referred to as scenario *S1* in the following) encompasses 20 service classes á 20 service candidates, 3 NFP (duration, waiting time, costs), and 5 users with 2 IUR each. For each setting we conduct 1,000 simulation runs and determine the average computation time (measured in milliseconds [ms]) Gurobi needs for solving each of both optimization model. For all simulation runs, we use a machine with an Intel Xeon E5-2470 v2 processor with 2.40 GHz, 32 GB RAM, Win7 64bit, Java 1.8, and Gurobi Optimizer 6.5.

Using our continuous time concept the computation time required for solving the problem *S1* is 120 ms, which holds true for all settings as the parameter change only concerns the discrete time approach. As the left diagram of Figure 2 illustrates, the discrete time approach needs much less than 120 ms for settings with a single-digit number of time intervals but – on the other side – more than 1,000 ms for settings with more than 100 time intervals. When conducting this experiment with other problem settings – e.g., different number of service classes (scenario *S2*), service candidates (scenario *S3*), users (scenario *S4*) or NFP – we achieve similar results (cf. Figure 2): The continuous approach is superior regarding computation time above a certain small number of time intervals. Therefore, in scenarios which would require a fine granular time concept (resulting in a high number of time intervals) – as it is the case with service re-selection at runtime – the continuous time concept greatly outperforms the discrete time concept in terms of computation time.

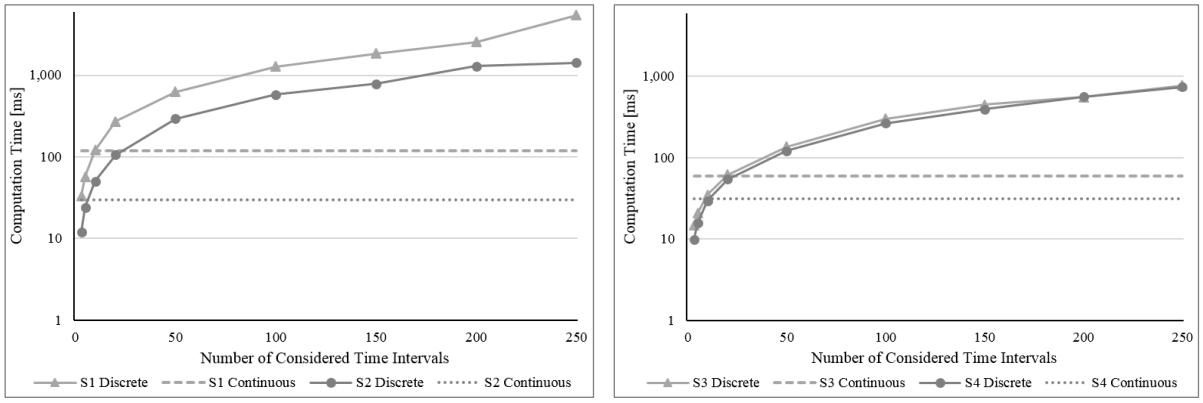

Figure 2. Performance evaluation of continuous vs. discrete time concept

5.2 Efficacy

For the evaluation of efficacy, we build upon a real-world scenario provided by Heinrich et al. (2015a): This scenario refers to a tourism city day trip by five individual persons (cf. Figure 3). In this context, services are understood as service objects (cf. Dannewitz et al., 2008; Hinkelmann et al., 2013) representing real-world entities (e.g., sight, museum, restaurant) that are determined by certain information services (TripAdvisor[3], GooglePlaces[4]). The considered process consists of 15 service classes or actions, respectively, and each action could be realized by a suitable service object – where a service object is characterized by its NFP *costs*, *recommendation value* and *duration*. Using the discrete time concept service objects with no fixed duration (e.g., restaurants, sights) would have to be integrated multiple times, each with a different possible manifestation of duration which have to fit to the considered discrete time intervals. As our approach allows for continuous consideration of time, we are able to leave it to the optimization model to determine the optimal duration of such a service object when solving the problem. More precisely, each of the five users has specified his/her individual target value for duration for each of the 15 actions – in addition to his/her personal weights and requirements regarding all NFP. Based on that, our problem setting encompasses 132 service objects allocated to the 15 actions of the

[3] http://www.programmableweb.com/api/tripadvisor, accessed September 2016

[4] http://www.programmableweb.com/api/google-places, accessed September 2016

process. Both *duration* and *waiting time* are considered as continuous variables according to our continuous time concept. Furthermore, each user participating in the process has defined three different IUR, which in total results in five mutual complementary, one mutual conflicting, eight simultaneous complementary, and one simultaneous conflicting IUR.

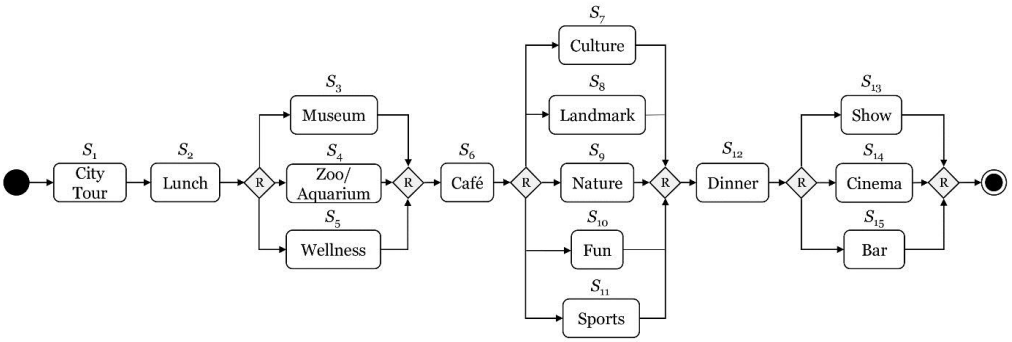

Figure 3. Process of the city day trip (Heinrich et al., 2015a)

During the city day trip there may occur various events that would require a re-selection of the initially planned service compositions. To demonstrate the efficacy of our approach we simulate the occurrence of three exemplary events and analyze the results:

a) Failing service (object): User 4 leaves the restaurant *Sababa* after 9 minutes waiting for a free table to have *2) lunch* elsewhere

b) User leaving the process: User 3 leaves the day trip after having coffee at the *6) café Livingroom*

c) Deviating NFP values: User 5's visit of the restaurant *Ringlers* for having *2) lunch* took 11 minutes longer than initially planned

For this purpose, we use our re-selection approach to obtain the new optimal service compositions of all users and compare them to the initially planned service compositions determined by the existing approach. Due to space restrictions, we focus in the following on discussing the differences in the initial vs. re-optimized service compositions for users 4 and 5 only.

Regarding event a), that means the "failure" of restaurant *Sababa* after 9 minutes, user 4 would not be able to have *2) lunch* without re-selection. After re-selection, the optimal solution for user 4 is having *2) lunch* at the restaurant *Ringlers* for 75 minutes. Since the 9 minutes at *Sababa* also count for the total duration of the day trip, the succeeding stay at *BMW World* (action *3) museum*) is reduced by this amount of time (from 75 min to 66 min) in order to fulfill user 4's constraint regarding the NFP duration. On the other hand, the failure of restaurant *Sababa* and the resulting switch to restaurant *Ringlers* enables the realization of an additional complementary IUR compared to the initially planned solution: "User 5 requests to visit *Pussers Bar* (action *15) bar*) simultaneously together with user 4".

As a consequence of b) with user 3 leaving the process, any IUR after action *6) café* with participation of user 3 could not be fulfilled any more, that means the initially expected positive utilities associated with complementary IUR are not realized but – on the other side – also the expected negative utilities related to conflicting IUR are avoided. For instance, this refers to the IUR "user 5 requests to visit *English Garden* (action *9) nature*) simultaneously with user 3".

Without re-selection, the occurrence of event c) (i.e. user 5's stay at *Ringlers* for *2) lunch* is increased by 11 minutes) would lead to a violation of user 5's global end-to-end constraint regarding NFP duration. Additionally, simultaneous complementary IUR planned further in the process would probably be not fulfilled, and simultaneous conflicting IUR expected to be avoided could potentially be realized. Indeed, in the re-optimized service composition user 5's visit of *3) museum BMW World* is abbreviated by 11 minutes to solve these issues. For example, as a result user 5's complementary IUR of visiting *English Garden* (action *9) nature*) simultaneously with user 3 can still be realized.

6 Conclusion, Limitations, and Further Research

Within this work, we presented a multi-user service re-selection approach enabling successful process recovery after occurrence of certain events at process runtime by determining the optimal service compositions for the remaining users and the remaining part of the process (contribution ❶).

For this, we proposed an optimization model taking into account multiple users with their preferences and requirements and dependencies resulting from mutual as well as simultaneous IUR. For the consideration of the latter, which requires temporal coordination of the users, we introduced a continuous time concept (contribution ❷) to overcome the flexibility and performance issues related to the existing discrete time concept – which is supported by the results of our performance evaluation in Section 5.1. We therefore contribute to the current body of knowledge in multi-user service (re-)selection. Furthermore, using the continuous time concept obviates the need for the definition of specific, discrete time intervals by the decision-maker when building the optimization model for a certain re-selection problem. Besides that, our findings also reveal important practical benefit. Depending on the problem size (i.e. number of service classes, service candidates and users), the number of possible service compositions can get extremely large. Additionally, with dependencies existing between different users and the required temporal coordination of the users' actions due to IUR this leads to a great complexity of the (initial) service selection problem. Thus, in case a process requires re-optimization due to an occurred event, the corresponding re-selection problem may still be of high complexity subject to the size of the remaining part of the process. The approach proposed in this work helps to deal with this complexity since it determines the optimal service composition for the rest of the process and each remaining user.

However, our approach is also subject to some limitations that need to be addressed in future research: As in this work the focus primarily lies upon the development of the model, we so far neglected the time-to-repair related to conducting re-selection at runtime (i.e. time needed from occurrence of an event until successful process continuation) (cf. Canfora et al., 2008; Sandionigi et al., 2013) – and by this also the duration overhead produced by the re-selection algorithm itself. From a practical point of view, the time-to-repair also needs to be taken into consideration in service re-selection as it adds up to the overall duration/ response time and also influences the realization of simultaneous IUR in the remaining part of the process. Furthermore, when aiming at an optimal solution as we do, we have to recognize that the service (re-)selection problem is NP-hard which generally corresponds to an exponential development in computation time (Abu-Khzam et al., 2015). In terms of service re-selection, this requires a cost-benefit tradeoff between improving the utility of a user's service composition and the duration overhead produced by the re-selection algorithm (Canfora et al., 2008). Thus, to further improve our approach we will in a next step focus on integrating the time-to-repair in our model and, additionally, on reducing the computation time needed for solving the multi-user service re-selection problem. A promising starting point for that could be the development of a heuristic technique (cf. e.g., Alrifai et al., 2012; Canfora et al., 2008; Qiqing et al., 2009) by means of an algorithm that efficiently scales with the problem size while achieving high decision quality, that means close-to-optimal solutions.

In conclusion, this work can serve as a first step for a comprehensive service re-selection approach regarding multi-user processes where dependencies among users exist, e.g., resulting from IUR.

References

Abu-Khzam, F. N., C. Bazgan, J. E. Haddad and F. Sikora (2015). "On the Complexity of QoS-Aware Service Selection Problem". In A. Barros (ed.) *Service-oriented computing*, pp. 345–352. Heidelberg: Springer.

Ai, L. and M. Tang (2008). "QoS-Based Web Service Composition Accommodating Inter-service Dependencies Using Minimal-Conflict Hill-Climbing Repair Genetic Algorithm". In: *IEEE Fourth International Conference on eScience*, pp. 119–126.

Alrifai, M. and T. Risse (2009). "Combining global optimization with local selection for efficient QoS-aware service composition". In: *Proceedings of the 18th International Conference on World Wide Web*. Ed. by J. Quemada, G. León, Y. Maarek, W. Nejdl, pp. 881–890.

Alrifai, M., T. Risse and W. Nejdl (2012). "A hybrid approach for efficient Web service composition with end-to-end QoS constraints" *ACM Transactions on the Web* 6 (2), 1–31.

Ardagna, D. and R. Mirandola (2010). "Per-flow optimal service selection for Web services based processes" *Journal of Systems and Software* 83 (8), 1512–1523.

Ardagna, D. and B. Pernici (2007). "Adaptive Service Composition in Flexible Processes" *IEEE Transactions on Software Engineering* 33 (6), 369–384.

Barry, D. K. (2012). *Web Services, Service-Oriented Architectures, and Cloud Computing. The Savvy Manager's Guide.* 2nd ed. Burlington: Elsevier Science.

Benouaret, K., D. Benslimane and A. Hadjali (2012). "Selecting Skyline Web Services for Multiple Users Preferences". In: *IEEE 19th International Conference on Web Services (ICWS)*, pp. 635–636.

Berbner, R., M. Spahn, N. Repp, O. Heckmann and R. Steinmetz (2007). "Dynamic Replanning of Web Service Workflows". In: *Inaugural IEEE-IES Digital EcoSystems and Technologies Conference*, pp. 211–216.

Canfora, G., M. Di Penta, R. Esposito and M. L. Villani (2008). "A framework for QoS-aware binding and re-binding of composite web services" *Journal of Systems and Software* 81 (10), 1754–1769.

Dannewitz, C., K. Pentikousis, R. Rembarz, É. Renault, O. Strandberg and J. Ubillos (2008). "Scenarios and research issues for a network of information". In *4th International Mobile Multimedia Communications Conference*.

He, Q., J. Han, Y. Yang, J. Grundy and H. Jin (2012). "QoS-Driven Service Selection for Multi-tenant SaaS". In: *IEEE 5th International Conference on Cloud Computing (CLOUD)*, pp. 566–573.

Heinrich, B., M. Klier, L. Lewerenz and M. Mayer (2015a). "Enhancing Decision Support in Multi User Service Selection". In: *36th International Conference on Information Systems (ICIS)*.

Heinrich, B., M. Klier, L. Lewerenz and S. Zimmermann (2015b). "Quality-of-Service-Aware Service Selection. A Novel Approach Considering Potential Service Failures and Nondeterministic Service Values" *Service Science* 7 (1), 48–69.

Hinkelmann, K., M. Maise and B. Thonssen (2013). "Connecting enterprise architecture and information objects using an enterprise ontology". In: *Enterprise Systems Conference (ES)*, pp. 1–11.

Jin, H., H. Zou, F. Yang, R. Lin and T. Shuai (2012). "A Novel Method for Optimizing Multi-User Service Selection" *Journal of Convergence Information Technology* 7 (5), 296–310.

Kang, G., J. Liu, M. Tang, X. Liu and K. K. Fletcher (2011). "Web Service Selection for Resolving Conflicting Service Requests". In: *IEEE International Conference on Web Services*, pp. 387–394.

Klusch, M. and P. Kapahnke (2012). "The iSeM matchmaker. A flexible approach for adaptive hybrid semantic service selection" *Web Semantics: Science, Services and Agents on the World Wide Web* 15, 1–14.

Lewis, J. and M. Fowler (2014). *Microservices*. URL: http://martinfowler.com/articles/microservices.html (visited on 14.01.16).

Li, J., D. Ma, L. Li and H. Zhu (2008). "AADSS. Agent-Based Adaptive Dynamic Semantic Web Service Selection". In: *4th International Conference on Next Generation Web Services Practices*: IEEE, pp. 83–89.

Li, J., D. Ma, X. Mei, H. Sun and Z. Zheng (2011). "Adaptive QoS-Aware Service Process Reconfiguration". In: *IEEE International Conference on Services Computing (SCC)*, pp. 282–289.

Lin, K.-J., J. Zhang, Y. Zhai and B. Xu (2010). "The design and implementation of service process reconfiguration with end-to-end QoS constraints in SOA" *Service Oriented Computing and Applications* 4 (3), 157–168.

Nemhauser, G. L. and L. A. Wolsey (1988). *Integer and combinatorial optimization.* New York: Wiley.

Papazoglou, M. P., P. Traverso, S. Dustdar and F. Leymann (2007). "Service-Oriented Computing. State of the Art and Research Challenges" *Computer* 40 (11), 38–45.

Prat, N., I. Comyn-Wattiau and J. Akoka (2015). "A Taxonomy of Evaluation Methods for Information Systems Artifacts" *Journal of Management Information Systems* 32 (3), 229–267.

Qiqing, F., P. Xiaoming, L. Qinghua and H. Yahui (2009). "A Global QoS Optimizing Web Services Selection Algorithm Based on MOACO for Dynamic Web Service Composition". In: *International Forum on Information Technology and Applications (IFITA)*, pp. 37–42.

Rodriguez-Mier, P., M. Mucientes and M. Lama (2012). "A Dynamic QoS-Aware Semantic Web Service Composition Algorithm". In C. Liu, H. Ludwig, F. Toumani and Q. Yu (eds.) *Service-oriented computing*, pp. 623–630. Berlin: Springer.

Sandionigi, C., D. Ardagna, G. Cugola and C. Ghezzi (2013). "Optimizing Service Selection and Allocation in Situational Computing Applications" *IEEE Transactions on Services Computing* 6 (3), 414–428.

Shen, Y., M. Wang, X. Tang, Y. Luo and M. Guo (2012a). "Context-Aware HCI Service Selection" *Mobile Information Systems* 8 (3), 231–254.

Shen, Y., X. Yang, Y. Wang and Z. Ye (2012b). "Optimizing QoS-Aware Services Composition for Concurrent Processes in Dynamic Resource-Constrained Environments". In: *IEEE 19th International Conference on Web Services (ICWS)*, pp. 250–258.

Sheng, Q. Z., X. Qiao, A. V. Vasilakos, C. Szabo, S. Bourne and X. Xu (2014). "Web services composition. A decade's overview" *Information Sciences* 280, 218–238.

Stein, S., T. R. Payne and N. R. Jennings (2009). "Flexible provisioning of web service workflows" *ACM Transactions on Internet Technology* 9 (1), 1–45.

Strunk, A. (2010). "QoS-Aware Service Composition: A Survey". In: *IEEE 8th European Conference on Web Services (ECOWS), 2010*. Ed. by A. Brogi. Piscataway, NJ: IEEE, pp. 67–74.

Wanchun, D., L. Chao, Z. Xuyun and J. Chen (2011). "A QoS-Aware Service Evaluation Method for Co-selecting a Shared Service". In: *IEEE International Conference on Web Services (ICWS)*, pp. 145–152.

Wang, H., J. Zhang, C. Wan, S. Shao, R. Cohen, J. Xu and P. Li (2010). "Web Service Selection for Multiple Agents with Incomplete Preferences". In: *IEEE/ACM International Conference on Web Intelligence-Intelligent Agent Technology*, pp. 565–572.

Weinhardt, C., B. Blau, T. Conte, L. Filipova-Neumann, T. Meinl and W. Michalk (2011). *Business Aspects of Web Services.* Berlin, Heidelberg: Springer.

Williams, H. P. (2013). *Model building in mathematical programming.* 5th Edition. Chichester: Wiley.

Yu, T. and K.-J. Lin (2005). "Adaptive algorithms for finding replacement services in autonomic distributed business processes". In: *Proceedings of Autonomous Decentralized Systems*, pp. 427–434.

Yu, T., Y. Zhang and K.-J. Lin (2007). "Efficient algorithms for Web services selection with end-to-end QoS constraints" *ACM Transactions on the Web* 1 (1).

Zeng, L., B. Benatallah, A. Ngu, M. Dumas, J. Kalagnanam and H. Chang (2004). "QoS-aware middleware for Web services composition" *IEEE Transactions on Software Engineering* 30 (5), 311–327.

Zheng, Z. and M. R. Lyu (2010). "An adaptive QoS-aware fault tolerance strategy for web services" *Empirical Software Engineering* 15 (4), 323–345.

Zheng, Z., Y. Zhang and M. R. Lyu (2014). "Investigating QoS of Real-World Web Services" *IEEE Transactions on Services Computing* 7 (1), 32–39.