QUALITY OF SERVICE-AWARE SERVICE SELECTION: A NOVEL APPROACH CONSIDERING POTENTIAL SERVICE FAILURES AND NON-DETERMINISTIC SERVICE VALUES

Bernd Heinrich, University of Regensburg
Mathias Klier, University of Regensburg
Lars Lewerenz, University of Regensburg
Steffen Zimmermann, University of Innsbruck

ABSTRACT

In service oriented systems, a variety of loosely coupled services are composed to support the execution of processes. One important research question in this context is, how such services can be selected by taking into account the values for the corresponding quality of service (QoS) attributes. Existing QoS-aware *ex-ante* service selection approaches build on preferences and requirements for the QoS attributes and evaluate the available services by means of a utility value. Due to many reasons like software bugs, high server workload or network failures an *ex-ante* optimal service may temporarily be unavailable or fail during its execution, requiring re-planning. Moreover, several QoS attribute values may be stochastic rather than deterministic. Thus, both the *ex-post* realized QoS values and the corresponding utility may significantly differ from the *ex-ante* computed ones, incurring a substantial loss of resources. In this paper we present a novel approach for the QoS-aware service selection considering both the effects of potential service failures and non-deterministic QoS values *ex-ante*. The approach is founded on an expected utility calculus. We find that considering the effects of potential service failures and non-deterministic QoS values leads to substantially better decisions in QoS-aware service selection.

Keywords: *QoS-aware service selection, IT services, service failures*

QUALITY OF SERVICE-AWARE SERVICE SELECTION: A NOVEL APPROACH CONSIDERING POTENTIAL SERVICE FAILURES AND NON-DETERMINISTIC SERVICE VALUES

INTRODUCTION

The service oriented architecture (SOA) paradigm has attracted much interest in the last decade. According to Forrester (2010) 84% of the global 2000 enterprises (Forbes 2010) currently use SOA and 70% of the SOA users plan to expand its use as a result of the perceived benefits. At the same time, the service market has taken a significant development. The service infomediary *ProgrammableWeb*, for instance, grew at an average annual rate of 161% between 2005 and 2014. Today, over 12,000 services from various categories (e.g. financial, enterprise, e-commerce, travel, government, etc.) are listed within the directory of *ProgrammableWeb*. The platform *AppExchange* by Salesforce offers over 1,100 services that can be integrated directly within the customers' CRM processes on Sales Cloud (cf. Weinhardt et al. 2011). The realization of SOA through the standardized technology of (web) services (Alrifai et al. 2012) as well as its characteristics (loose coupling, dynamic binding, open standards, simplicity, and security) (Erl 2009) open up the possibility of versatile and flexible ad-hoc collaborations between different business partners (Luthria and Rabhi 2009; Ren and Lyytinen 2008). Companies can use a service, for instance, to verify and correct customer addresses during their order processes (cf. Informatica 2014). Moreover, the standardized interfaces of services do not only allow for the realization of single functionalities through single services but also enable the composition of several services (Alrifai et al. 2012; Ardagna and Pernici 2007). Such service compositions can realize more complex functionalities through to whole business processes (e.g. the *PayPal* service composed with the *DocuSign* service allows users to sign documents and collect payments at the same time (DocuSign 2014)) (cf. AbuJarour and Awad 2014; Medjahed et al. 2014; Moghadda and Davis 2014; Weise et al. 2014). For instance, Weinhardt et al. (2011) discuss an online payment process

where each process activity (e.g. validate customer data, payment transaction, and customer data storage) is realized by a service (e.g. *Informatica* – validate customer data, *CyberSource* – payment transaction, and *Amazon* – data storage). In the course of a growing service market more and more functional equivalent services are provided (e.g. credit card validation, flight booking, hotel search, online payment, communication, etc.) which, however, differ in their so called quality of service (QoS) attributes such as availability, response time, or price. For instance, to verify customer data the service market comprises several services by different providers (e.g. *ServiceObjects*, *CDYNE*, *Informatica*, and *PostcodeAnywhere*). All of them provide the same functionality but each with different values for the QoS attributes[1]. Under the premise that the service market keeps on growing, the challenge will not only be to find services that realize the required functionalities but also to select the best services out of functional equivalent services while considering their QoS values (Kritikos et al. 2014; Zeng et al. 2004). This results in an optimization problem which has to consider the following two aspects: First, the best service composition (i.e. the one with the highest utility) regarding the QoS values of the included services has to be selected. Second, the QoS values of this service composition need to satisfy given QoS requirements (e.g. an upper limit regarding the end-to-end response time of the service composition).

In literature this optimization problem is widely known as QoS-aware service selection which aims at determining the optimal service composition *ex-ante* (i.e. before starting to execute the services). Several approaches have been proposed to address this problem (Alrifai et al. 2012; Ardagna and Pernici 2007; Canfora and Di Penta et al. 2005; Yu et al. 2007; Zeng et al. 2004; Zheng and Xiong 2012). However, during the runtime of the process there are situations where an ex-ante selected service candidate is not available (i.e. the execution of a service candidate fails). These runtime

[1] For instance, with respect to price per call (per August 2, 2014): *ServiceObjects*: $ 0.079; *CDYNE*: $ 0.035; *Informatica* AddressDoctor: $ 0.50; *PostcodeAnywhere*: $ 0.08.

failures may result from, notably, communication infrastructure exceptions, failures in the operation in the middleware, server overload, or incorrect input parameter types (cf. Chan et al. 2007). Service failures require the re-planning of the service composition during the runtime of the process (cf. Canfora et al. 2008; Lin et al. 2010; Zeng et al. 2004) in order to ensure that the process can still be executed. Besides service failures, there are also situations where the QoS values realized during the runtime of the process deviate from those planned ex-ante. These deviations result from the fact that several QoS values are not deterministic but rather stochastic over time (Fakhfakh et al. 2012; Hwang et al. 2007; Stein et al. 2009). Existing approaches, however, neither consider potential service failures nor non-deterministic QoS values in the QoS-aware *ex-ante* service selection which lead to the following effects:

❶ Due to re-planning, the QoS values realized ex-post and thus, the *realized utility* of a service composition, may *significantly differ* from the ex-ante planned ones (Canfora et al. 2008). This effect occurs, for instance, when service candidates fail and have to be replaced by other service candidates with different QoS values (e.g. execution price or response time).

❷ In case of re-planning, the *feasibility* of service compositions regarding the given QoS requirements may be affected as a result of the hitherto realized QoS values.

❸ It takes a certain *time for the failure of a service to be noticed and compensated*, which is comparable to the time-to-repair (Hwang et al. 2007; Mani and Nagarajan 2002; Maximilien and Singh 2004). This time interval has a direct influence on the end-to-end response time and thus on the utility of the affected service composition.

❹ In case of a service failure and a re-planning that switches to an alternative service composition, *losses* can occur if services that have already been executed are not part of the alternative service composition. These losses directly influence the end-to-end QoS values and thus the utility of the service composition.

❺ In case of non-deterministic QoS values the *realized utility* can *significantly differ* from the ex-

ante planned one. Deviations of QoS values can cause higher consumptions than expected with respect to QoS attributes and even violations of QoS requirements (e.g. due to outliers) during the runtime of the service composition (Berbner et al. 2007; Canfora et al. 2008).

As a result of the effects ❶-❺ the optimal ex-ante selected service composition may significantly differ from the optimal ex-post one after process execution. Neglecting these effects in the QoS-aware ex-ante service selection may cause an unnecessary waste of resources (e.g. time and money) and may prevent the process from being executed under the given conditions (cf. QoS requirements).

Hence, we propose a novel approach for the QoS-aware ex-ante service selection where both the effects of potential service failures (effects ❶-❹) and non-deterministic QoS values (effect ❺) are considered ex-ante. We find that considering these effects usually leads to better decisions. Thus, the proposed approach can help save resources and allows for a better QoS-aware ex-ante service selection compared to existing approaches.

The paper is structured as follows: The next section discusses the literature related to the QoS-aware ex-ante service selection problem. Afterwards, we introduce a running example (according to Canfora et al. 2008) that is used to illustrate the problem of the QoS-aware ex-ante service selection as well as for evaluation purposes later on. In the third section, our approach is presented in two steps. First, we propose an analytical model to consider the effects ❶-❹. In the second step, taking effect ❺ into account, we present a simulation model. To point out the strength of our approach compared to existing approaches, we provide a mathematical evaluation in the fourth section. In addition, the applicability and the benefits of our approach are demonstrated by means of an example. Finally, we conclude with a discussion of important limitations and an outlook on future research.

RELATED LITERATURE

Our research directly contributes to the literature on (1) QoS-aware ex-ante service selection and is related to the literature on (2) QoS-aware fault tolerant strategy selection and on (3) QoS-aware re-planning of service compositions.

The literature on (1) aims at determining the optimal QoS-aware service composition ex-ante and comprises several analytical (cf. Ardagna and Pernici 2007; Cui et al. 2011; Huang et al. 2009; Yu et al. 2007; Zeng et al. 2004) and heuristic approaches (cf. Alrifai et al. 2012; Ardagna and Mirandola 2010; Berbner et al. 2006; Canfora and Di Penta et al. 2005; Canfora et al. 2008; Jaeger and Muehl 2007; Li and Yan-xiang 2012; Maolin and Ai 2010; Menascé et al. 2008; Yang et al. 2010; Yu et al. 2007; Zheng and Xiong 2012)[2]. The QoS-aware ex-ante service selection is realized either by a local or a global optimization. Within the local optimization, the utility of a single service candidate is calculated based on its values for the QoS attributes. To determine the optimal QoS-aware service composition, exactly one service candidate per service class[3] is selected that has the highest utility value and satisfies the local QoS requirements (Alrifai et al. 2012; Zeng et al. 2004). Concerning the global optimization, the values of each QoS attribute are first aggregated for a specific service composition (end-to-end QoS value) and finally weighted to calculate an overall utility value for this service composition (Ardagna and Pernici 2007; Canfora et al. 2008; Yu et al. 2007). To conduct the local or global optimization, in most cases (Alrifai et al. 2012; Ardagna and Pernici 2007; Zeng et al. 2004) Multiple Criteria Decision Making (Hwang and Yoon 1981) is applied to determine the optimal QoS-aware service composition while satisfying the (local or global) QoS requirements.

In all of these QoS-aware *selection approaches* potential service failures are only considered by the QoS attribute availability. Aggregating the availabilities of the single service candidates

---

[2] A detailed discussion of these approaches can be made available by the authors upon request.

[3] A service class is defined as a set of services that offer the same functionality but may differ in their QoS values.

included in a service composition the availability of the whole service composition is obtained. Finally, this end-to-end QoS value is used in combination with the other end-to-end QoS attribute values (e.g. response time or price) to determine the utility of the service composition. However, representing service failures solely by the QoS attribute availability ignores their effects ❶-❹ on other QoS attributes such as response time, and thus on the utility of a service composition. Hence, a special treatment of potential service failures is required. We contribute to the literature on QoS-aware ex-ante service selection by considering the effects of potential service failures in combination with the effect ❺ of non-deterministic QoS values.

The approaches on (2) aim to select the optimal QoS-aware fault tolerant strategy for the execution of service candidates (Stein et al. 2009; Zheng and Lyu 2010). For instance, Zheng and Lyu (2010) propose strategies to either retry the execution of the failed service candidate for a predefined number of times (retry), to execute different service candidates in sequential order if the primary service candidate fails (round robin), or to execute functional equivalent service candidates in parallel (active). Based on the selected strategy they either obtain lower execution costs with a higher response time or vice versa. Stein et al. (2009) propose quite similar strategies including a parallel and a sequential strategy but further combine them in a flexible manner. Additionally, they consider non-deterministic QoS values for the response time of service candidates. The approaches on (2) use the probability of failure (i.e. 1-availability) of a service candidate to select a suitable fault tolerant strategy. In case a service candidate has, for instance, a high probability of failure the parallel execution of functional equivalent service candidates becomes beneficial. This means that redundancy and higher costs are accepted to avoid a long response time in case a service candidate fails. The approaches on (2) do not focus on the QoS-aware ex-ante service selection or on the determination of the optimal service composition. However, they are nevertheless related to our research and provide interesting insights into strategies for coping with the QoS attribute availability. In particular, the use of the probabilities of failure in decision making seems promising for the QoS-aware ex-ante service selection as well as for the consideration of the effects of

potential service failures on other QoS attributes (e.g. response time or price).

In contrast to (1), the literature on (3) aims at QoS-aware re-planning approaches (cf. Ardagna and Pernici 2007; Berbner et al. 2007; Canfora et al. 2008; Li et al. 2011; Lin et al. 2010; Yu and Lin 2005; Zeng et al. 2004) to consider service failures when they actually emerge (i.e. during the runtime of the process). There are three types of approaches: First, those that re-optimize the remaining part of a service composition after every single service invocation (cf. Berbner et al. 2007). Second, those that try to substitute the failed service candidate with an alternative service candidate and expand the number of service candidates that are substituted if no feasible substitution can be found (cf. Ardagna and Pernici 2007; Li et al. 2011; Lin et al. 2010). Third, approaches that determine and re-optimize the remaining part of the service composition that has to be executed if a service candidate fails (cf. Berbner et al. 2007; Canfora et al. 2008; Yu and Lin 2005; Zeng et al. 2004). In all cases, the re-planning is based upon deterministic QoS values. We discuss the approaches on (3) since they provide some interesting insights into how the failure of a service candidate can be compensated during the *runtime of the process*. In this case, the approaches either substitute the faulty service candidate with another feasible service candidate from the same service class or they switch to another feasible service composition. Knowing these options is also necessary for the appropriate consideration of the effects of potential service failures within the QoS-aware ex-ante service selection.

To conclude this section, we are not aware of any existing approach that addresses the effects ❶-❺ (or even one of the effects ❶-❹) within a QoS-aware ex-ante service selection. Thus, we extend the approaches in (1) by considering the effects ❶-❺. This allows for the approaches on (2) and (3) to be used as knowledge base.

INTRODUCTION OF A RUNNING EXAMPLE

Our running example is based upon a travel booking process (cf. Figure 1) and was initially developed as part of a project for a service oriented marketplace (cf. Canfora et al. 2005; Canfora et al. 2008). We use this example due to the following reasons: First, the travel domain has practical relevance. *ProgrammableWeb*, for instance, offers more than 200 services for this domain. This may also be a reason why the travel domain is widely used to illustrate the problem of QoS-aware service selection (cf. Cardellini et al. 2007; Chifu et al. 2010; Dai et al. 2009; Gao et al. 2006; Grossmann et al. 2011; Guo et al. 2007; Hwang et al. 2008; Li et al. 2010; Li et al. 2011; Mei et al. 2008; Yang et al. 2009; Zeng et al. 2008). Second, a documentation of the service candidates and their corresponding QoS values for the QoS attributes response time and price is provided by Canfora et al. (2008). Due to the fact that no information is included concerning the QoS attribute availability, we used a service monitoring tool[4] to complement the example with historical values for the availability of similar service candidates.

After introducing the running example, we apply an existing selection approach to determine the optimal QoS-aware services ex-ante. Detailed information regarding the notation used (cf. Yu et al. 2007; Zeng et al. 2004) is provided in the Appendix (cf. Table 4).

Example of a Travel Booking Process (according to Canfora et al. 2008)

The first task of the travel booking process is to search for available flights as well as for free accommodation near the address provided by the user. For that purpose either the service classes S_1-S_3 or the service class S_4 can be used (cf. pick construct, e.g. Wan et al. 2008, Yu et al. 2007). Hence, compositions of service candidates of the service classes S_1-S_3 and service candidates of the service class S_4 are functionally equivalent. Based on the arrival time and the latest possible hotel check-ins (cf. S_5) either information about shuttle prices (in 80% of the cases) or taxi prices

[4] http://monitor.programmableweb.com/ (accessed in 01/2015)

from the airport to the hotel (in 20% of the cases) are provided (cf. S_6 or S_7). The distance from the chosen hotel to a specific tourist attraction is calculated (cf. S_8) in parallel. Based on this calculation, either information about car rental (in 10% of the cases) or metro card prices (in 90% of the cases) are presented (S_9 or S_{10}). The corresponding service candidates s_{ij} of each service class S_i (with i=1...10; j=1...3) and their QoS values are summarized in the Appendix (cf. Table 5). The end-to-end QoS requirement with respect to the response time is 40,000 ms.

- Insert Figure 1. around here -

Exemplified QoS-aware ex-ante service selection according to existing approaches

In the following, we apply an existing analytical selection approach to the example above to illustrate how the QoS-aware ex-ante service selection problem is solved. For this purpose and without loss of generality (i.e. other analytical approaches provide the same solution), we applied the approach by Yu et al. (2007) called MCSP. MCSP is based upon the multiconstrained optimal path problem and the shortest path problem in acyclic directed graphs. This approach was chosen, although it has a lower runtime performance than other analytical approaches (cf. Ardagna and Pernici 2007; Zeng et al. 2004), as with small adaptions the second, third, fourth, and so on best[5] service compositions could easily be determined as well – a fact that is especially important with respect to our evaluation conducted later on. In a first step, the *possible* service compositions are determined. The whole process (cf. Figure 1) involves ten service classes S_i, each with three service candidates $s_{i\,1}$, $s_{i\,2}$ and $s_{i\,3}$ (with i=1 to 10). To determine all possible service compositions considering the workflow structures we use the idea of execution routes (cf. Alrifai et al. 2012; Ardagna and Pernici 2007; Yu et al. 2007; Zeng et al. 2004). An execution route is defined as a path from the start to the end of the process which contains all branches of each parallel split and

[5] Thereby it is possible to determine the service compositions which are feasible subject to the given end-to-end QoS requirements and rank them according to their utility values.

only one branch of each XOR split. Overall eight possible execution routes have to be considered (cf. Table 1).

In total there are 9,724 *possible* service compositions to execute the process. The number of *feasible* service compositions depends on the given end-to-end QoS requirements. To determine the optimal service composition out of the feasible ones, the utility function *U* is applied, which is defined as follows:

$$U(s_{ij}) = \sum_{\propto=1}^{x} w_\alpha * \left(\frac{q_{ij}^{\propto}}{\max\limits_{i,j} q_{ij}^{\propto} - \min\limits_{i,j} q_{ij}^{\propto}} \right) + \sum_{\beta=1}^{y} w_\beta * \left(\frac{-q_{ij}^{\beta}}{\max\limits_{i,j} q_{ij}^{\beta} - \min\limits_{i,j} q_{ij}^{\beta}} \right) \quad (1)^6$$

Considering $U(s_{ij})$ there are *x* QoS attributes (with α=1 to *x*) that have to be maximized (e.g. availability) and *y* QoS attributes (with β=1 to *y*) that have to be minimized (e.g. response time). $q_{ij}^{\propto}$ and q_{ij}^{β} represent the QoS values for service candidate s_{ij} and QoS attributes α and β, respectively. The user can set up preferences (weights w_α , w_β) for each QoS attribute ($0<w_\alpha, w_\beta<1$ and $\sum_{\alpha=1}^{x} w_\alpha + \sum_{\beta=1}^{y} w_\beta = 1$). In our example, all QoS attributes (response time, price, and availability) have the same weight of $\frac{1}{3}$. Same as Alrifai et al. (2012), the different QoS values are normalized with the distance between the maximum and the minimum value of a QoS attribute over all service classes S_i (with i=1 to *I*) and service candidates s_{ij} (with j=1 to J_i). This is done to prevent a selection approach from being biased by the scaling of the QoS values.

Using MCSP by Yu et al. (2007) and the utility function $U(s_{ij})$ (cf. term 1) the following optimal QoS-aware service compositions are determined for the execution routes (cf. Table 1).

Table 1. Optimal QoS-aware service compositions per execution route

[6] A similar utility function can be found in Alrifai et al. (2012), Ardagna and Pernici (2007) and Zeng et al. (2004). However, in contrast to these works we decided to take a slightly different approach to normalize the QoS values. This way, we ensure the same utility value regardless of whether it is determined based on the aggregated QoS values of the whole service composition or based on the sum of utility values of the single service candidates that are element of the corresponding service composition. In addition, this adaptation does not affect utility-based rankings of single services and service compositions, respectively.

No.	Execution Route	Optimal Service composition	Price	Response Time	Availability	Utility
1	S_1-S_2-S_3-S_5-S_6-S_8-S_9	$s_{1\ 2}$-$s_{2\ 1}$-$s_{3\ 2}$-$s_{5\ 2}$-$s_{6\ 2}$-$s_{8\ 1}$-$s_{9\ 1}$	10.77	17,900	0.666	-1.981
2	S_4-S_5-S_6-S_8-S_9	$s_{4\ 1}$-$s_{5\ 2}$-$s_{6\ 2}$-$s_{8\ 1}$-$s_{9\ 1}$	12.75	19,900	0.708	-2.090
3	S_1-S_2-S_3-S_5-S_6-S_8-S_{10}	$s_{1\ 2}$-$s_{2\ 1}$-$s_{3\ 2}$-$s_{5\ 2}$-$s_{6\ 2}$-$s_{8\ 1}$-$s_{10\ 3}$	6.77	17,400	0.619	-1.800
4	S_4-S_5-S_6-S_8-S_{10}	$s_{4\ 1}$-$s_{5\ 2}$-$s_{6\ 2}$-$s_{8\ 1}$-$s_{10\ 3}$	8.75	19,400	0.658	-1.909
5	S_1-S_2-S_3-S_5-S_7-S_8-S_9	$s_{1\ 2}$-$s_{2\ 1}$-$s_{3\ 2}$-$s_{5\ 2}$-$s_{7\ 1}$-$s_{8\ 1}$-$s_{9\ 1}$	10.72	17,900	0.681	-1.944
6	S_4-S_5-S_7-S_8-S_9	$s_{4\ 1}$-$s_{5\ 2}$-$s_{7\ 1}$-$s_{8\ 1}$-$s_{9\ 1}$	12.70	19,900	0.724	-2.053
7	S_1-S_2-S_3-S_5-S_7-S_8-S_{10}	$s_{1\ 2}$-$s_{2\ 1}$-$s_{3\ 2}$-$s_{5\ 2}$-$s_{7\ 1}$-$s_{8\ 1}$-$s_{10\ 3}$	6.72	17,400	0.632	-1.764
8	S_4-S_5-S_7-S_8-S_{10}	$s_{4\ 1}$-$s_{5\ 2}$-$s_{7\ 1}$-$s_{8\ 1}$-$s_{10\ 3}$	8.70	19,400	0.672	-1.872

Focusing for instance on execution route 2, the optimal QoS-aware service composition is determined to $s_{4\ 1}$-$s_{5\ 2}$-$s_{6\ 2}$-$s_{8\ 1}$-$s_{9\ 1}$ with an end-to-end price of 12.75, an end-to-end response time of 19,900, and an end-to-end availability of 0.708. The results of the existing analytical approaches in Table 1 serve as a reference base for the evaluation of our approach later on.

NOVEL APPROACH CONSIDERING THE EFFECTS OF POTENTIAL SERVICE FAILURES

First, we present an analytical model where the effects of potential service failures (effects ❶-❹) are addressed. Second, we broaden the problem context by considering non-deterministic QoS values (effect ❺) and propose a simulation model. In correspondence with existing approaches (cf. Alrifai et al. 2012; Ardagna and Pernici 2007; Yu et al. 2007), both models focus on the selection of the optimal QoS-aware service candidates per execution route of the considered process (cf. running example above).

Analytical Model

The basic idea of our approach is to consider the effects of potential service failures by means of an expected utility determined for a service candidate and subsequently for a whole service composition. To achieve this, the QoS attribute availability (represented by a probability) is used to weight the utility that is realized in case the respective service candidate is available while the failure rate (represented by the counter probability) is used to weight the utility that is realized in case the service candidate fails (cf. effect ❶). In the latter case, the time interval until a service failure is noticed and compensated (cf. effect ❸) and potential losses (cf. effect ❹) are taken into

account when determining the expected utility. Furthermore, the (expected) end-to-end QoS values of a service composition can be calculated including the effects of service failures and then verified according to their feasibility with respect to the QoS requirements (cf. effect ❷).

Based on the notation summarized in the Appendix (cf. Table 4) our optimization model is defined as follows:

$$\max_{x_{ij}} \sum_{S_i \in \Upsilon} \sum_{s_{ij} \in S_i} \mathrm{E}\left[U(s_{ij}), U_R(s_{ij}), p_{ij}\right] * x_{ij}$$

$$\text{Subject to: } \underset{S_i \in \Upsilon,\, s_{ij} \in S_i}{\Phi^n} \left(\mathrm{E}\left[q_{ij}^n, q_R^n(s_{ij}), p_{ij}\right] * x_{ij}\right) \leq Q_c^n \qquad \forall n = 1, .., N$$

$$\sum_{s_{ij} \in S_i} x_{ij} = 1\,;\ \forall S_i \in \Upsilon\,;\ x_{ij} \in \{0; 1\} \tag{2}$$

Considering the service classes S_i included in execution route Υ as well as the respective service candidates $s_{ij} \in S_i$, the optimization model determines for a risk neutral decision maker the decision variables x_{ij} (x_{ij}=1 indicates that service candidate s_{ij} is selected; x_{ij}=0 that it is not) to maximize the accumulated expected utility of the selected service candidates. For each service class S_i exactly one service candidate s_{ij} has to be selected. At the same time, the aggregated expected QoS values of the service composition need to satisfy the end-to-end QoS requirements $Q_c = [Q_c^1, \ldots, Q_c^N]^T$[7] for every QoS attribute *n* (with *n*=1 to *N*). This means that the expected QoS values $\mathrm{E}\left[q_{ij}^n, q_R^n(s_{ij}), p_{ij}\right]$ aggregated (with Φ^n as aggregation function) over all service candidates s_{ij} included in the service composition need to be less than or equal to Q_c^n[8] ($\forall n$=1 to *N*). Please notice that in case functional equivalent execution routes exists, the service candidates of the

[7] For QoS attributes that have to be maximized (e.g. reputation) the corresponding constraint has to be multiplied with minus one so that it holds that the aggregated QoS values need to be less than the given QoS requirements.

[8] Notice: Depending on the intended analysis, when determining the aggregated QoS values of a service composition (cf. term (2)) the deterministic QoS values q_{ij}^n may be used instead of the expected values $\mathrm{E}\left[q_{ij}^n, q_R^n(s_{ij}), p_{ij}\right]$ considering a potential re-planning.

execution route creating the highest accumulated expected utility among all functional equivalent execution routes have to be selected for the execution of the process.

In the optimization model, the major challenge is to consider the effects of potential service failures when determining the expected utility $\mathrm{E}\left[U(s_{ij}), U_R(s_{ij}), p_{ij}\right]$ of a service candidate $s_{ij} \in S_i, S_i \in \Upsilon$. A service candidate is only available with probability p_{ij} but fails with probability $(1 - p_{ij})$. Thus, when determining $\mathrm{E}\left[U(s_{ij}), U_R(s_{ij}), p_{ij}\right]$, the utility $U(s_{ij})$ is weighted with the probability p_{ij}, whereas the expected utility $U_R(s_{ij})$ is weighted with the probability $(1 - p_{ij})$. Here, $U_R(s_{ij})$ represents the maximum of the expected utilities of the possible re-planning options $\mathrm{E}[\dots]$ (in case service candidate s_{ij} fails). In the following, these re-planning options and the respective calculations are discussed in general and illustrated by means of our running example. For reasons of comprehensibility, we decided to use an excerpt of the running example (cf. Figure 2) and focus on the QoS attribute response time as well as on the expected utility of a service candidate s_{ij}. Moreover, as the response time is the only QoS attribute considered we leave out the normalization in the denominator of the utility function provided by term (1). Nevertheless, the calculations can analogously be conducted for other QoS attributes and the whole process. In the following, we analyze a *potential failure of service candidate* $s_{3\,1}$ considering the following general re-planning options.

- Insert Figure 2. around here -

1. Option 1: Select the next best service candidate $s_{ij'}$ which belongs to the same service class S_i and is feasible subject to the QoS requirements Q_c: For that purpose the expected utility and the expected QoS values of every service candidate $s_{ij'} \in S_i$ (with $s_{ij'} \neq s_{ij}$) need to be calculated. Moreover, the time interval until the failure of service candidate s_{ij} is noticed and compensated (cf. effect ❸) needs to be considered. We suppose the expected value of this time

interval to be $\frac{t_{ij}}{2}$[9] (with t_{ij} representing the response time of service candidate s_{ij}). Consequently, the service candidate $s_{ij'}$ creating the highest expected utility among all alternative service candidates will be selected.

For a better understanding of this re-planning option we analyze a *potential failure of service candidate* $s_{3\,1}$ in a situation where the service candidate $s_{1\,1}$ of the preceding service class S_1 is available (cf. Figure 2). Here, the expected utility of the alternative service candidate $s_{3\,2}$ has to be determined as follows (cf. Figure 3): First, for the case that service candidate $s_{3\,2}$ is *available* (upper path in Figure 3), besides the utility of service candidate $s_{3\,2}$ (-2,000) the expected utility that results in regard to the best service candidate in the succeeding service class S_5 needs to be taken into account[10]. This means that the expected utilities of the service candidates $s_{5\,1}$ and $s_{5\,2}$ have to be determined and compared considering a possible termination of the process (cf. option 3b) resulting in an expected utility of -50,000 as well. For service candidate $s_{5\,1}$ it is calculated to $\mathrm{E}[\ldots] = 0.97 * (-4{,}500) + 0.03 * (-2{,}250 + (-5{,}257)) = -4{,}590.21$. The first summand represents that service candidate $s_{5\,1}$ is available. In the second summand (i.e. service candidate $s_{5\,1}$ is not available) the time interval until the failure of service candidate $s_{5\,1}$ is noticed and compensated has to be considered with $\frac{t_{ij}}{2} = \frac{4{,}500}{2} = 2{,}250$. Moreover, the service candidate $s_{5\,2}$ has to be taken into account as re-planning option resulting in an expected utility of $\mathrm{E}[\ldots] = 0.98 * (-4{,}300) + 0.02 * (-2{,}150 + (-50{,}000)) = -5{,}257$. Here, it is also considered that after a potential failure of service candidate $s_{5\,2}$ the execution of the process needs to be terminated

[9] Supposing the time interval until the failure is noticed and compensated to be uniformly distributed between 0 and t_{ij}, the expected value is given by $\frac{t_{ij}}{2}$.

[10] As S_5 is the last service class of the process (cf. Figure 2) no further succeeding service classes have to be considered.

(cf. option 3a; 'inevitable Termination' resulting in an expected utility of -50,000) as no further alternative service candidates are available. Analogous calculations for service candidate $s_{5\,2}$ result in an expected utility of -4,375.65. Thus, in case $s_{3\,2}$ is available, service candidate $s_{5\,2}$ is determined as the best service candidate of the succeeding service class S_5.

Second, for the case that the alternative candidate $s_{3\,2}$ *fails* (lower path in Figure 3), service candidate $s_{4\,1}$ is determined as the best re-planning option (compared to option 3b 'optional Termination') due to its expected utility (cf. Figure 3 – ellipse labeled with "Re-Planning") which is analogously calculated to $E[\dots] = 0{,}82 * (-8{,}500 + \text{Max}[-4{,}590.21; -4{,}375.65; -50{,}000]) + 0.18 * (-4{,}250 + (-50{,}000)) = -20{,}323.03$. To conclude, the expected utility of service candidate $s_{3\,2}$, representing re-planning option 1, is calculated to $E[\dots] = 0.92 * (-2{,}000 + \text{Max}[-4{,}590.21; -4{,}375.65; -50{,}000]) + 0.08 * (-1{,}000 + \text{Max}[-50{,}000; -20{,}323.03]) = -7{,}571.44$.

- Insert Figure 3. around here -

2. Option 2: Select the service candidate $s_{i'j'}$ in the next best alternative service composition avoiding service class S_i which is feasible subject to the QoS requirements Q_c. For this option, the expected utility and the expected QoS values of the service candidates in the alternative service compositions need to be calculated. The service candidate $s_{i'j'}$ creating the highest expected utility among the alternative service compositions and which is feasible subject to the QoS requirements will be selected. Aside from the consideration of the time interval until the service failure is noticed and compensated, any emerging losses (cf. effect ❹) also need to be taken into account within option 2. Losses emerge in situations where service candidates initially intended for the execution of the process are not part of the alternative service composition.

   For a better understanding of this re-planning option, we again analyze a *potential failure* of service candidate $s_{3\,1}$ in a situation where service candidate $s_{1\,1}$ of the preceding service class

S_1 is available (cf. Figure 2). Thus, we focus on service candidate $s_{4\,1}$ when analyzing re-planning option 2 (cf. Figure 4). The expected utility of service candidate $s_{4\,1}$ can be determined in a very similar way compared to re-planning option 1 and is calculated to $E[\dots] = 0.82 * (-8{,}500 + \text{Max}[-4{,}590.21; -4{,}375.65; -50{,}000]) + 0.18 * (-4{,}250 + \text{Max}[-50{,}000; -9{,}945.60]) = -13.113{,}24$. Note that if service candidate $s_{4\,1}$ is available here, the QoS values of service candidate $s_{1\,1}$ (implicitly) constitute losses as this service candidate is not part of the alternative service composition (cf. Figure 2).

- Insert Figure 4. around here -

3. Option 3: Terminate the execution of the process: In this case, disutility needs to be appointed (e.g. data loss as a result of a process termination (AWS Team 2012) or the emergence of business costs (Kieninger et al. 2013) caused by a service failure).
   a) This option is *inevitable* in situations where no feasible alternative service candidate within the same service class of the faulty service candidate (cf. option 1) and no feasible alternative service composition avoiding the service class S_i (cf. option 2) can be found.
   b) A termination of process execution is also *beneficial* in situations where the expected utility of a feasible re-planning (cf. option 1 or 2) is lower than the expected utility of an immediate termination. This is the case if alternative service candidates or service compositions are feasible but with respect to the corresponding utility worse compared to the (dis)utility resulting from an immediate process termination. The same holds in case the service composition needs to be terminated later on (due to reasons mentioned above). Here, a re-planning is not economically worthwhile as further service candidates would be executed and further resources would be consumed, although the process has to be terminated anyway.

In our example, there is an option for a 'beneficial Termination' (cf. option 3b) in case of a *potential failure of service candidate* $s_{3\,1}$. This option results in an expected utility of -50,000.

Moreover, Figure 3 and Figure 4 also illustrate the calculation and consideration of the expected utility of the re-planning option ‘Termination’ for our example.

After calculating the expected utility for each re-planning option, the option with the highest expected utility is selected. This expected utility is multiplied with the probability $(1 - p_{ij})$ representing the case that service candidate s_{ij} fails. The utility $U(s_{ij})$ is multiplied with the probability p_{ij} representing the case that service candidate s_{ij} is available. Hence, the expected utility $\mathrm{E}\left[U(s_{ij}), U_R(s_{ij}), p_{ij}\right]$ of service candidate s_{ij} is given by:

$$\mathrm{E}\left[U(s_{ij}), U_R(s_{ij}), p_{ij}\right] = U(s_{ij}) * p_{ij} + U_R(s_{ij}) * (1 - p_{ij}) \quad (3)$$

The corresponding expected QoS values $\mathrm{E}\left[q_{ij}^n, q_R^n(s_{ij}), p_{ij}\right]$ for each attribute *n* can be derived accordingly based on the QoS value q_{ij}^n, the expected QoS value $q_R^n(s_{ij})$ in case of a failure, and the availability p_{ij} of service candidate s_{ij}.

In the example above (cf. Figure 2), re-planning option 1 provides the highest expected utility and thus would be selected in case of a potential failure of service candidate $s_{3\,1}$. Hence, the expected utility of service candidate $s_{3\,1}$ is calculated to $\mathrm{E}\left[U(s_{3\,1}), U_R(s_{3\,1}), p_{3\,1}\right] =$ -2,500 * 0.9 + (-1,250 + (-7,571.44))* (1 - 0.9) = -3,132.14.

The discussion above shows that each service candidate is not only evaluated based on its own utility respective QoS values (e.g. the utility of -2,500 for service candidate $s_{3\,1}$ in the example). Rather, the effects (resulting from the re-planning options) in case of a failure of this service candidate (e.g. the utility of (-1,250 + (-7,571.44)) in case service candidate $s_{3\,1}$ fails) are considered as well.

To conclude, with the expected utility $\mathrm{E}\left[U(s_{ij}), U_R(s_{ij}), p_{ij}\right]$ the optimal service candidates $s_{ij} \in S_i, S_i \in \Upsilon$ can ex-ante be selected (cf. term (2)) while considering the following effects of potential service failures.

❶ By using expected utility calculus the effects of potential service failures can now be considered within the QoS-aware ex-ante service selection (i.e. the target function).

❷ The aggregated (expected) end-to-end QoS values allow for an ex-ante consideration of the effects of potential service failures on the feasibility of service compositions.

❸ Temporal delays until a service failure is noticed and compensated are considered when determining both the expected utility and the aggregated (expected) end-to-end QoS values.

❹ Losses that can occur due to a re-planning are taken into account. These losses influence the expected utility and the aggregated (expected) end-to-end QoS values of a service composition and thus its valuation compared to other service compositions.

With the proposed approach, we aim to make better decisions in the QoS-aware ex-ante service selection. In this sense the waste of resources like time and money can be mitigated or prevented.

Simulation Model

In the following we extend the analytical model by relaxing the assumption that all QoS values are deterministic. This is especially necessary for QoS values that are non-deterministic by nature such as response time or reliability (Fakhfakh et al. 2012; Hwang et al. 2007; Stein et al. 2009). Figure 5 depicts the real-world values (N=6,537) for the QoS attribute response time of the web service DOTS address validation[11].

- Insert Figure 5. around here -

It illustrates that considering non-deterministic QoS values (cf. effect ❺) is crucial as operating with a single deterministic QoS value does not reflect the real-world. The use of probability distributions for the QoS values leads to several challenges when determining the optimal service

[11] Service provided by serviceobjects.com (http://www.serviceobjects.com/support/performance-reports accessed in 1/2015).

composition. To address these challenges it is favorable to define a simulation model due to the following reasons:

- *Determination of the end-to-end QoS values*

  Using probability distributions for the QoS values may imply that aggregating them to the end-to-end QoS values of a service composition is not promising in an analytical way. Following from this, the validation of the QoS requirements is no longer practical in an analytical way either. This problem can be illustrated by means of an example with only two service candidates that are executed sequentially. Even if the response times of these service candidates follow different log normal distributions (cf. approx. the distribution shown in Figure 5) it is not practical to determine the resulting end-to-end QoS value for the response time by means of a probability distribution in closed form. This is especially true the bigger (e.g. more service classes and service candidates per service class) and the more complex (e.g. different probability distributions) the problem is. In these cases the use of numeric techniques such as simulations is more favorable.

- *Consideration of different workflow structures*

  Considering non-deterministic QoS values in combination with different workflow structures can lead to situations where the aggregation of QoS values is not promising in an analytical way. We will briefly illustrate this, considering a parallel split-synchronization structure and the response time as a QoS attribute: If the response times of two service candidates that are executed in parallel are represented by random variables (e.g. Z_1 and Z_2) following normal distributions with different expected values and variances, $\mathrm{E}\left[Max[Z_1, Z_2]\right] \neq Max\left[\mathrm{E}[Z_1]; \mathrm{E}[Z_2]\right]$ holds. Hence, the determination of the probability distribution of the

aggregated response times in closed form is not practical[12], which leads to the same problems as above.

Thus, a simulation model is introduced to cope with effects ❶-❺. Similarly to the analytical model, the idea of the simulation model is to determine the expected utility of the service candidates and subsequently for a whole service composition. Thus, in the simulation model, we iteratively analyze each service composition $sc \in SC$ representing a tuple of service candidates, with exactly one service candidate for each service class of the considered execution route Υ (SC represents the set of all possible service compositions for execution route Υ). When analyzing the service candidates s_{ij} of service composition sc ($s_{ij} \in sc, sc \in SC$), the effects ❶-❺ are considered accordingly.

The non-determinism of the QoS values is modelled by using random variables for each QoS attribute n (with n=1 to N). $\widetilde{Q_{ij}} = \left[\widetilde{Q_{ij}^1}, \ldots, \widetilde{Q_{ij}^N}\right]^T$ represents the QoS vector for a service candidate s_{ij} including all random variables $\widetilde{Q_{ij}^n}$ (with i=1 to I, j=1 to J, and n=1 to N) following the probability distribution $F_Q(q_{ij}^n)$. To determine the expected utility $\mathrm{E}\left[U(s_{ij}), U_R(s_{ij}), p_{ij}\right]$ of a service candidate $s_{ij} \in sc$, its execution is simulated for a predefined number of simulation runs M. This means that if a service candidate s_{ij} is included in service composition sc, a realization q_{ij}^n for each random variable $\widetilde{Q_{ij}^n}$ has to be drawn in each simulation run based on the corresponding probability distribution $F_Q(q_{ij}^n)$. To consider potential service failures within the simulation experiment, every service candidate s_{ij} is available with probability p_{ij} and fails with probability

[12] There are approaches that provide methods for the aggregation of non-deterministic QoS values in an analytical way (e.g. Fakhfakh et al. 2012; Hwang et al. 2007). But in contrast to the selection approach presented in this paper, those approaches take a runtime perspective. Hence, based on the invoked services, they know under certainty which values for the QoS attributes are realized and thus are able to determine the end-to-end QoS values in a closed form solution. For the ex-ante selection of services, however, those approaches are not applicable as it is not known which values for the QoS attributes will be realized based on the corresponding probability distribution.

$(1 - p_{ij})$. To address this, we draw a realization d of the random variable $\widetilde{D}$ which follows a uniform distribution on the interval [0; 1]. Starting with the first service candidate s_{ij} of a service composition, this is accomplished by comparing the realization d with the probability of availability p_{ij} (this is done for all subsequent service candidates of the service composition as well). More precisely two cases can be distinguished:

- *Case a*: $d \leq p_{ij}$ represents that service candidate s_{ij} is available in the simulated execution.
- *Case b*: $d > p_{ij}$ represents that service candidate s_{ij} fails in the simulated execution.

Case a: As service candidate s_{ij} is available, the realizations q_{ij}^n of the random variables $\widetilde{Q_{ij}^n}$ are aggregated with the already processed QoS values for each simulation run. Here, the aggregation of the QoS values depends on the workflow structures (e.g. parallel-split, synchronization, XOR-split, and simple merge) of the process (Ardagna and Pernici 2007; Canfora et al. 2008; Yu et al. 2007). A detailed description to handle further workflow patterns (e.g. loop) can be found in Fakhfakh et al. (2012), Huang et al. (2009) and Hwang et al. (2007). Afterwards, it has to be verified whether the service composition is still feasible[13], as the aggregated realizations (non-deterministic QoS values) could cause a violation of the QoS requirements. As a result two cases can emerge.

- *Case a.1:* $\phi^n(q_{ij}^n) \leq Q_r^n \; \forall n$=1 to *N*. The aggregated QoS values of all QoS attributes satisfy the corresponding QoS requirements in the simulation run. In this case, the next service candidate of the considered service composition is analyzed.
- *Case a.2:* $\exists\, n$: $\phi^n(q_{ij}^n) > Q_r^n$. At least one aggregated QoS value of a QoS attribute does not satisfy the corresponding QoS requirement in the simulation run. Here, a re-planning is

[13] This can be evaluated using different procedures Berbner et al. (2007), Canfora et al. (2008).

necessary to ensure feasibility of the considered service composition (cf. case *b* below).

Case b: In case *b* either a service candidate fails during its simulated execution or at least one of the aggregated QoS values does not satisfy the corresponding QoS requirement. Here, we have to analyze different re-planning options, in which the unavailable service candidate will not be considered. This holds until the next simulation run m+1 is started. The re-planning options are analogous to the analytical model and will therefore not be discussed in more detail. The following two cases can result:

- *Case b.1:* There is at least one feasible service candidate or service composition that allows a further process execution (cf. option 1 and 2) and results in a higher utility compared to the termination of the process.
- *Case b.2:* There is no feasible alternative service candidate or service composition that allows a further process execution (option 3a), or a termination of the process is beneficial as the expected utility of a feasible re-planning (cf. option 1 or 2) is lower than the expected utility of an immediate termination (option 3b). In this case the execution of the service composition has to be terminated and the corresponding (dis)utility is processed.

With the help of the simulation model, every service candidate s_{ij} can be evaluated regarding its expected utility. Over the total number of simulation runs M of a simulation experiment, the average expected utility $\mathrm{E}\left[U(s_{ij}), U_R(s_{ij}), p_{ij}\right]$ of a service candidate s_{ij} can be determined. As a result, each service candidate s_{ij} and thus each service composition sc can be evaluated in order to select the optimal QoS-aware services ex-ante while considering the effects of potential service failures (cf. effects ❶-❹) as well as non-deterministic QoS values (cf. effect ❺). The simulation model and the re-planning procedure are illustrated in Nassi-Schneiderman diagrams in the Appendix (cf. Figure 8 and Figure 9).

EVALUATION

In this section we show that our analytical model leads – under certain conditions – to better results compared to existing QoS-aware ex-ante service selection approaches. If these conditions are not met, the results of the analytical model coincide with those of existing approaches providing an exact solution. Afterwards, we demonstrate the applicability of both the analytical model and the simulation model by means of the running example introduced above.

Evaluation of the Analytical Model

In the following, we state two findings. Together, these findings show that our approach allows for better decision making.

FINDING 1: *Considering a feasible alternative service candidate (cf. option 1) and the effects of potential service failures (especially effects ❶ and ❸) our approach leads – under certain conditions – to better ex-ante decisions for the QoS-aware service selection compared to existing selection approaches.*

FINDING 2: *Considering a feasible alternative service composition (cf. option 2) and the effects of potential service failures (especially effects ❷ and ❹) our approach leads – under certain conditions – to better ex-ante decisions for the QoS-aware service selection compared to existing selection approaches.*

Both findings can be demonstrated with the help of basic selection problems (it is obvious that the findings also hold for more complex problems involving further service candidates, QoS attributes, etc.)[14]. Even based on the basic selection problems it can be shown that if the effects of potential service failures (cf. effects ❶ to ❹) are neglected this can – under prevalent conditions (i.e. these conditions are not special cases) – lead to wrong ex-ante decisions. Under these conditions, determining and using expected utilities and expected QoS values as proposed by our approach is

[14] Proofs can be made available by the authors upon request.

therefore beneficial regarding the QoS-aware ex-ante service selection.

Findings 1 and 2 explicate that compared to existing selection approaches our approach indeed allows for a better ex-ante decision making, which is reflected in the following aspects:

(1) Compared to existing approaches our approach considers alternative service candidates (cf. option 1) and service compositions (cf. option 2) and evaluates them with respect to their expected utility including the effects of potential service failures (cf. effect ❶).
(2) The provided approach takes into account that service failures may affect the feasibility of alternative service candidates and service compositions (cf. re-planning options) with respect to the QoS requirements (cf. effect ❷).
(3) The time interval until service failures are noticed and compensated (cf. effect ❸) and losses that may occur in the course of re-planning (cf. effect ❹) are considered as well.

Demonstration of the Applicability

The goal of this evaluation step is to demonstrate the applicability of our approach and to show that this approach can lead to better results compared to existing selection approaches. In terms of better results we do not aim to provide a runtime optimized approach or a heuristic. It is rather about the question of how the effects resulting from potential service failures in combination with non-deterministic QoS values can be considered in a well-founded way resulting in better decisions in QoS-aware service selection. To show that our approach is manageable with respect to its computation time, an evaluation of its performance is provided at the end of this section.

We divided this evaluation step into two subsections. In the first, we evaluate the analytical model with respect to effects ❶-❹. In the second, we evaluate the simulation model with respect to effects ❶-❹ and ❺. This split into two subsections is useful to obtain comparability between current selection approaches and our approach, since current analytical selection approaches do not consider non-deterministic QoS-values. Having demonstrated that our approach considering the effects of potential service failures ❶-❹ allows for better decision making, we are then able to

additionally evaluate the provided approach with respect to effect ❺.

To ensure transparency and reproducibility, we again use the example according to (Canfora et al. 2008) and the utility function (cf. Alrifai et al. 2012; Ardagna and Pernici 2007; Zeng et al. 2004) given in equation (1). To illustrate that our results are feasible for different execution routes in the example, we exemplarily focus on execution route 2 in the first subsection and on execution route 1 in the second subsection, respectively. However the results for all other execution routes are very similar and summarized in the Appendix (cf. Table 6 and Table 7).

Analytical model

Using existing selection approaches, the optimal QoS-aware service composition for execution route 2 is determined to $s_{4\,1}$-$s_{5\,2}$-$s_{6\,2}$-$s_{8\,1}$-$s_{9\,1}$ with an end-to-end response time of 19,900, an end-to-end price of 12.75, and an end-to-end availability of 0.708 (cf. Table 1). In contrast, when applying our analytical model the optimal QoS-aware service composition is $s_{4\,3}$-$s_{5\,2}$-$s_{6\,1}$-$s_{8\,3}$-$s_{9\,3}$.

Table 2. Proposed approach vs. existing selection approaches (execution route 2; deterministic QoS values)

Service composition	Results based on existing selection approaches			Results based on the proposed approach			Rank order	
	Resp. Time	Price	Avail.	*Exp.* Resp. Time	*Exp.* Price	*Exp.* Utility	**Existing approaches**	**Proposed approach**
$s_{4\,1}$-$s_{5\,2}$-$s_{6\,2}$-$s_{8\,1}$-$s_{9\,1}$	19,900	12.75	0.708	23,831	17.34	-3.008	1	116
$s_{4\,1}$-$s_{5\,2}$-$s_{6\,3}$-$s_{8\,1}$-$s_{9\,1}$	19,900	12.60	0.701	23,971	17.48	-3.019	2	123
$s_{4\,1}$-$s_{5\,2}$-$s_{6\,1}$-$s_{8\,1}$-$s_{9\,1}$	19,900	12.70	0.701	23,773	16.89	-2.947	3	86
$s_{4\,3}$-$s_{5\,2}$-$s_{6\,1}$-$s_{8\,3}$-$s_{9\,3}$	24,800	10.80	0.645	27,401	12.78	-2.709	110	1
$s_{4\,3}$-$s_{5\,2}$-$s_{6\,2}$-$s_{8\,1}$-$s_{9\,1}$	18,400	14.05	0.717	22,292	15.35	-2.713	6	2
$s_{4\,3}$-$s_{5\,2}$-$s_{6\,1}$-$s_{8\,3}$-$s_{9\,1}$	20,800	12.80	0.658	24,679	14.24	-2.714	84	3

As the results in Table 2 illustrate, service composition $s_{4\,1}$-$s_{5\,2}$-$s_{6\,2}$-$s_{8\,1}$-$s_{9\,1}$ which is determined as the optimal one by existing selection approaches is only in 116th position when the effects of potential services failures ❶-❹ are considered. More precisely, if we analyze how these effects influence the end-to-end QoS values and thus the utility of the service composition $s_{4\,1}$-$s_{5\,2}$-$s_{6\,2}$-$s_{8\,1}$-$s_{9\,1}$ we come to the following results: Although this service composition has a lower probability of re-planning (=1-probability of availability=1-0.708=0.292) than service composition $s_{4\,3}$-$s_{5\,2}$-

$s_{6\,1}$-$s_{8\,3}$-$s_{9\,3}$ (=0.355), its end-to-end QoS values are more influenced (absolutely and relatively) by the effects of potential service failures. For example, the end-to-end price of service composition $s_{4\,1}$-$s_{5\,2}$-$s_{6\,2}$-$s_{8\,1}$-$s_{9\,1}$ increases from 12.75 to 17.34 (about 36%) which is two times higher compared to service composition $s_{4\,3}$-$s_{5\,2}$-$s_{6\,1}$-$s_{8\,3}$-$s_{9\,3}$ (about 18%). The same holds for the QoS attribute response time. This means that potential service failures have a much greater effect on the utility of service composition $s_{4\,1}$-$s_{5\,2}$-$s_{6\,2}$-$s_{8\,1}$-$s_{9\,1}$ than on service composition $s_{4\,3}$-$s_{5\,2}$-$s_{6\,1}$-$s_{8\,3}$-$s_{9\,3}$. To demonstrate the effects of service failures on the QoS values of a service composition in more detail, we use the following excerpt of our example.

- Insert Figure 6. around here -

Here, we focus on the effects that the potential failure of service candidate $s_{4\,1}$ has on the end-to-end QoS value response time. The probability that service candidate $s_{4\,1}$ fails is 0.19 (cf. Table 5 in the Appendix). Regarding this failure and the calculations using the analytical model, a re-planning is successful with probability 0.16 (with probability 0.03 the process has to be prematurely terminated). In this case, the expected response time is 35,792, which is – compared to the end-to-end response time of 19,900 (cf. Table 1) without considering the effects of potential service failures – an increase of about 80%. Thus, our approach proposes to select – amongst others – service candidate $s_{4\,3}$ (here the increase of the response time in case of a successful re-planning is only 42%) instead of service candidate $s_{4\,1}$.

Another reason why service composition $s_{4\,1}$-$s_{5\,2}$-$s_{6\,2}$-$s_{8\,1}$-$s_{9\,1}$ (which is selected by existing approaches) is worse compared to other service compositions can be found in its bad robustness with respect to service failures. Indeed, the probability of a premature termination of this service composition is more than twice as high compared to service composition $s_{4\,3}$-$s_{5\,2}$-$s_{6\,1}$-$s_{8\,3}$-$s_{9\,3}$. These results clearly indicate that the presented approach does not only help to save resources but also increases the chances of successfully executing the process. Further, they also illustrate that a special treatment of the QoS attribute availability considering the effects of potential service

failures is absolutely necessary.

Simulation model

With respect to the simulation model we first illustrate the importance to consider the effects ❶-❺ (*results of the simulation model*). Afterwards, we demonstrate that the computation time to solve the simulation model is manageable (*performance of solving the simulation model*).

Results of the simulation model

To evaluate the simulation model, the non-determinism of the QoS attribute response time[15] is focused in the following (cf. effect ❺). We prototypically implemented the simulation model and conducted our evaluation with the following parameterization: The execution of a service composition sc was simulated M=10,000 times in order to determine the expected utility $\mathrm{E}\left[U(s_{ij}), U_R(s_{ij}), p_{ij}\right]$ of the service candidates $s_{ij} \in sc$. Thereby, we used log-normal distributed random variables $\widetilde{Q_{ij}^T}$ (cf. real-world values of the QoS attribute in Figure 5) that are characterized by an expected value $\mathrm{E}\left[\widetilde{Q_{ij}^T}\right]$[16] and a standard deviations of 10% with respect to the expected value. To illustrate effect ❺ we compare the results determined by our approach with versus without considering non-deterministic QoS values. These results show a difference in the optimal QoS-aware service composition that is determined for each case. This holds true for all execution routes of the process (cf. Table 7 in the Appendix). In the following we exemplarily focus on the results for execution route 1.

Table 3 illustrates that considering only the effects of potential service failures service composition $s_{1\,2}$-$s_{2\,1}$-$s_{3\,1}$-$s_{5\,2}$-$s_{6\,2}$-$s_{8\,3}$-$s_{9\,2}$ is determined as the optimal one; this service composition is, however,

---

[15] As the price is not that volatile in most realistic cases, for illustration purposes we decided to focus on the response time and left the price deterministic.

[16] To determine the expected values $\mathrm{E}\left[\widetilde{Q_{ij}^T}\right]$ we used the QoS values for the response time (deterministic case) given in Table 5 in the Appendix.

only in 27th position when taking the non-determinism of the QoS attribute response time into account as well. In contrast, considering this non-determinism, service composition $s_{1\ 2}$-$s_{2\ 1}$-$s_{3\ 3}$-$s_{5\ 2}$-$s_{6\ 1}$-$s_{8\ 2}$-$s_{9\ 3}$ is determined as the optimal one (only in 114th position when using deterministic QoS values).

Table 3. Our approach with vs. without considering non-deterministic QoS values (execution route 1)

Service composition	Results based on deterministic QoS			Results based on non-deterministic QoS			Rank order	
	Exp. Resp. Time	*Exp.* Price	*Exp.* Utility	*Exp.* Resp. Time	*Exp.* Price	*Exp.* Utility	**Deter. QoS**	**Non-deter. QoS**
$s_{1\,2}$-$s_{2\,1}$-$s_{3\,1}$-$s_{5\,2}$-$s_{6\,2}$-$s_{8\,3}$-$s_{9\,2}$	29,127	8.18	-2.312	29,342	9.82	-2.485	1	27
$s_{1\,2}$-$s_{2\,1}$-$s_{3\,1}$-$s_{5\,2}$-$s_{6\,3}$-$s_{8\,1}$-$s_{9\,3}$	25,913	9.84	-2.315	26,136	11.23	-2.468	2	39
$s_{1\,2}$-$s_{2\,1}$-$s_{3\,3}$-$s_{5\,2}$-$s_{6\,2}$-$s_{8\,3}$-$s_{9\,1}$	25,978	9.91	-2.324	26,002	11.77	-2.524	3	120
$s_{1\,2}$-$s_{2\,1}$-$s_{3\,3}$-$s_{5\,2}$-$s_{6\,1}$-$s_{8\,2}$-$s_{9\,3}$	28,933	9.11	-2.386	29,109	9.40	-2.440	114	1
$s_{1\,2}$-$s_{2\,1}$-$s_{3\,3}$-$s_{5\,2}$-$s_{6\,3}$-$s_{8\,1}$-$s_{9\,3}$	26,547	9.88	-2.359	26,425	10.88	-2.450	42	2
$s_{1\,2}$-$s_{2\,1}$-$s_{3\,3}$-$s_{5\,1}$-$s_{6\,2}$-$s_{8\,1}$-$s_{9\,2}$	27,878	9.12	-2.338	28,070	10.13	-2.451	17	3

Based upon these findings, we obtain two important insights: First, the results provide evidence for the argumentation presented in the previous section. The response time of the parallel split-synchronization structure (S_1, S_2 and S_3) is 8,000 for service composition $s_{1\,2}$-$s_{2\,1}$-$s_{3\,1}$-$s_{5\,2}$-$s_{6\,2}$-$s_{8\,3}$-$s_{9\,2}$ based on deterministic QoS values, while the expected value of the response time is 8,392 when non-deterministic QoS values are used (this effect is also observed for other service compositions). To determine the expected value of 8,392 we approximated the frequency distribution of the simulated response times concerning the parallel split-synchronization structure (i.e. the service candidates $s_{1\,2}$, $s_{2\,1}$ and $s_{3\,1}$). The difference of 392 in the response time is due to the fact that for non-deterministic QoS values situations exist, where the maximum response time of the parallel split-synchronization structure results either from the realized response times of service candidates $s_{1\,2}$ and $s_{3\,1}$ or from the realized response time of service candidate $s_{2\,1}$ (i.e. $\mathrm{E}\left[Max[Z_1, Z_2]\right] \neq Max\left[\mathrm{E}[Z_1]; \mathrm{E}[Z_2]\right]$). Hence, to aggregate the QoS values properly and subsequently to select the optimal QoS-aware services ex-ante, the use of numeric techniques (e.g. a simulation model) is reasonable. Second, as our results indicate (cf. Table 3 above and Table 7 in the Appendix) the use of non-deterministic QoS values does have an impact on the QoS-aware ex-ante service selection. An explanation can be found in the increased number of premature process terminations due to service failures in combination with the use of non-deterministic QoS values. This number does not increase to the same extent for every service composition. In the

example, the absolute increase for service composition $s_{1\ 2}$-$s_{2\ 1}$-$s_{3\ 2}$-$s_{5\ 2}$-$s_{6\ 2}$-$s_{8\ 1}$-$s_{9\ 3}$ is 73 (total number of premature process terminations 175) whereas for service composition $s_{1\ 2}$-$s_{2\ 1}$-$s_{3\ 1}$-$s_{5\ 2}$-$s_{6\ 2}$-$s_{8\ 3}$-$s_{9\ 2}$ the absolute increase is just 23 (total number of premature process terminations 145). In this case, the use of service composition $s_{1\ 2}$-$s_{2\ 1}$-$s_{3\ 2}$-$s_{5\ 2}$-$s_{6\ 2}$-$s_{8\ 1}$-$s_{9\ 3}$ leads to a higher loss of resources and thus to a lower expected utility compared to the service composition $s_{1\ 2}$-$s_{2\ 1}$-$s_{3\ 1}$-$s_{5\ 2}$-$s_{6\ 2}$-$s_{8\ 3}$-$s_{9\ 2}$. Thus, capable of considering the effects of potential service failures and additionally non-deterministic QoS values, our simulation model determined the service composition $s_{1\ 2}$-$s_{2\ 1}$-$s_{3\ 3}$-$s_{5\ 2}$-$s_{6\ 1}$-$s_{8\ 2}$-$s_{9\ 3}$ as the optimal one.

Concluding remark: In all of the eight execution routes the QoS-aware service composition determined by current selection approaches is inferior compared to the optimal service compositions when considering the effects of potential service failures. This holds true, although the given QoS requirements are not very restrictive[17]. Obviously, the effects of potential service failures and the use of non-deterministic QoS values will even have a greater impact on the QoS-aware ex-ante service selection in case the QoS requirements are more restrictive.

Performance of solving the simulation model

With respect to the practical applicability it is further necessary to evaluate the simulation model in terms of its computation time. The latter depends on the runtime complexity of the selection and re-planning approach used and the number of service compositions sc considered. The runtime complexity of the selection and re-planning approach is influenced by the size of the search space which depends on the number of service classes I and the number of service candidates per service class J_i. For our performance evaluation we use the process based upon our running example. In the basic setting the number of service classes I and the number of service candidates per service

[17] For instance, as the QoS requirement for response time was set to 40,000, all 9,724 possible service compositions are feasible.

class J_i is 5 and the number of service compositions sc considered is 50. Then we extended this basic setting in three different scenarios:

- Scenario I: The number of service candidates per service class J_i is increased in steps of 5 from 5 to 50.
- Scenario II: The number of service classes I is increased in steps of 5 from 5 to 50.
- Scenario III: The number of service compositions sc considered is increased in steps of 50 from 50 to 500.

We simulated each scenario 20 times and calculated the average computation time. All analyses were conducted on a machine with an Intel Core I7 processor with 3.6 GHz, 16 GB RAM, and Java 1.8. The results are shown in Figure 7.

- Insert Figure 7. around here -

Figure 7 a) and b) illustrate that the computation time increases overproportionally depending on the size of the search space. These results were expected (cf. Alrifai et al. 2012; Yu et al. 2007) as the selection problem is modeled as multi-dimensional multi-constrained knapsack problem which is known to be NP-hard (cf. Martello and Toth 1987). However, even for an increased search space (e.g. I=50, J_i=5, Scenario II) the average computation time is 157 seconds. Since our purpose was not to present a computation time optimized selection approach or a heuristic but rather a first approach to determine the optimal service composition ex-ante considering the effects of potential service failures as well as non-deterministic QoS values, these computation times seem manageable. Figure 7 c) illustrates that the computation time increases in a linear way depending on the number of service compositions sc considered. In this respect, our simulation model shows a good scalability.

CONCLUSION, LIMITATIONS AND FUTURE RESEARCH

In this paper we address the QoS-aware service selection that aims to determine ex-ante the optimal service composition. This optimization problem is intensively discussed in the literature. However, the effects of potential service failures in combination with non-deterministic QoS values have not been considered by existing selection approaches yet.

Our approach is thought to contribute to address this research gap (cf. effect ❶-❺). Such an approach is highly relevant in cases where a planned service candidate is no longer available or may fail during its execution. Moreover, we argue that several QoS values are not deterministic but rather stochastic over time. Thus, neglecting the effects of potential service failures in combination with the non-determinism of several QoS attributes can lead to a significant waste of resources. To develop this approach we use expected utility calculus in combination with probability distributions to allow for a methodologically well-founded decision making in the QoS-aware ex-ante service selection. Indeed, we find that considering ex-ante the effects of potential service failures leads to better decisions. This could be shown in our evaluation. Furthermore, we illustrate the strengths and benefits of our approach by means of an example. The findings indicate that this approach outperforms existing service selection approaches.

Our results provide important managerial implications. First, decision makers should be aware of the significant effects that potential service failures as well as non-deterministic QoS values can have on the utility of a service composition and thus on the used resources. As demonstrated by our example, these effects can be considerable. Indeed, we observed cases where the difference between the end-to-end QoS values with versus without considering the effects of potential service failures was up to 80%. Thus, practitioners should be aware that there might be a huge gap between the (planned) end-to-end QoS values (cf. time and money) before starting the execution of a process and those actually realized afterwards. Furthermore, our findings also indicate that our approach improves the chances of successfully completing the execution of a service supported

process (cf. evaluation section). This may be an important factor for critical processes. Given the positive impacts, we believe that practitioners would substantially benefit from using the approach presented here when selecting service candidates for a process.

Moreover, we have to discuss the limitations which are the starting point for future research: First, the *expected utility* is a reliable decision criterion if the process and thus the service composition are executed many times. Then, the expected value is a very good estimator for the realized mean value ("law of large numbers"). This has to be taken into account when applying the approach. Second, regarding the *non-deterministic QoS values* we used random variables and probability distributions. This implies that information about these probability distributions is known. One common way to determine these probability distributions is the use of historical data (i.e. realized QoS values). This data can be obtained either directly from service providers (cf. e.g. Serviceobjects 2014) or with the help of monitoring tools[18]. Moreover, where intra-company services are also involved in the process, the tracking and monitoring can be realized by the use of service management tools (e.g. IBM WebSphere Integration Developer). Nevertheless, there are situations (e.g. a recently offered service candidate) where determining the probability distributions based on historical data is not possible. Here, further research is necessary. Third, in line with the current QoS-aware service selection approaches (cf. Alrifai et al. 2012; Ardagna and Pernici 2007; Zeng et al. 2004), we considered the case that service failures are *independent* of each other. However, in practice there are situations where a failure of a single service candidate may be accompanied by failures of other service candidates as well. In case a service provider faces a server overload problem, for example, it is very likely that besides the initially failed service further services of this particular provider are neither available. Indeed, our simulation model

[18] Cf. http://monitor.programmableweb.com/ (accessed in 01/2015) or http://www.keynote.com/solutions/monitoring/web-monitoring/ (accessed in 01/2015)

would generally be capable of considering such service interdependencies[19]. However, to maintain comparability with the existing QoS-aware service selection approaches we decided not to incorporate service interdependencies in this paper. In this way the importance of considering the effects of potential service failures could be illustrated in a more focused and objective way. Nevertheless, the simulation model presented here can serve as a basis to consider service interdependencies in the future as well. In this respect, however, further research is necessary on how to obtain the respective data needed (e.g. distributions, correlations, and frequencies of failure types). Fourth, future work is intended to support the further assessment and justification in different real-use situations.

A further goal for research is to analyze how existing heuristics (e.g. Alrifai et al. 2012) can be combined with our ideas to consider expected utilities, losses, etc. In the example above, but also in larger cases with more service classes and service candidates, the runtime of the optimization using our approach is still practical. Yet, in very large and complex cases – with numerous service classes and service candidates and many complex, nested workflow structures – heuristics are useful. However, the goal of this paper is not to provide a runtime optimized approach or a heuristic. It is rather about the question of how the effects resulting from potential failures of services in combination with non-deterministic QoS values can be considered in a well-founded way. In future work, we will work on how existing heuristics for the QoS-aware ex-ante service selection can be combined with our ideas.

We want to conclude with a more general perspective on the generalizability and the breadth of the application of our approach. We have illustrated that our approach is appropriate for the selection of web services which have the same functionality but differ in their QoS values. Because

[19] Let us consider a situation in which a service candidate fails due to a server overload problem. This situation could be considered by setting the availability of the remaining service candidates, having the same provider as the failed one, to zero. As a result, those service candidates will not be considered in the current re-planning and the simulation run *m*.

of the latter characteristic, we expect the proposed approach to be transferable to other IT services (e.g. mobile services). Here, service selection also comprises the consideration of functional equivalent services which are characterized by different QoS values and have to be composed in order to support the execution of a business process. The approach presented here can serve as a promising and well-founded basis for further research in this interesting field.

Acknowledgements

The research was funded by the Austrian Science Fund (FWF): P 23546-G11 and P 23567-G11.

References

AbuJarour M, Awad A (2014) Web Services and Business Processes: A Round Trip. Bouguettaya A, Sheng QZ, Daniel F, eds. Web Service Foundations (Springer, New York), 3–31.

Alrifai M, Risse T, Nejdl W (2012) A Hybrid Approach for Efficient Web Service Composition with End-to-End QoS Constraints. ACM Trans. Web 6(2):1–31.

Ardagna D, Mirandola R (2010) Per-flow Optimal Service Selection for Web Services Based Processes. Journal of Systems and Software 83(8):1512–1523.

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

AWS Team (2012) Summary of the Amazon EC2 and Amazon RDS Service Disruption in the US East Region, http://aws.amazon.com/de/message/65648/, accessed January 15, 2015.

Berbner R, Spahn M, Repp N, Heckmann O, Steinmetz R (2006) Heuristics for QoS-aware Web Service Composition. IEEE Computer Society, ed. Proceedings of the 2006, IEEE International Conference on Web Services. (Chicago, Illinois), 72–82.

Berbner R, Spahn M, Repp N, Heckmann O, Steinmetz R (2007) Dynamic Replanning of Web Service Workflows. IEEE Computer Society, ed. Proceedings of the 2007, IEEE International Conference on Digital EcoSystems and Technologies. (Cairns, Australia), 211–216.

Canfora G, Corte P, Nigro AD, Desideri D, Penta MD, Esposito R, Falanga A, Scognamiglio R, Torelli F, Villani ML, Zampognaro P (2005) The C-Cube Framework: Developing Autonomic Applications Through Web Services. Garlan D, Litoiu M, Müller HA, Mylopoulos J, Smith DB, Wong K, eds. Proceedings of the 2005, International Workshop on Design and Evolution of Autonomic Application Software. (St. Louis, Missouri), 1–6.

Canfora G, Di Penta M, Esposito R, Villani ML (2005) An Approach for QoS-aware Service Composition Based on Genetic Algorithms. Beyer H, ed. Proceedings of the 2005, Conference on Genetic and Evolutionary computation. (Washington, DC), 1069–1075.

Canfora G, Di Penta M, Esposito R, Villani ML (2008) A Framework for QoS-aware Binding and Re-binding of Composite Web Services. Journal of Systems and Software 81(10):1754–1769.

Cardellini V, Casalicchio E, Grassi V, Presti FL (2007) Flow-Based Service Selection for Web Service Composition Supporting Multiple QoS Classes. IEEE Computer Society, ed. Proceedings of the 2007, IEEE International Conference on Web Services., 743–750.

Chan MKS, Bishop J, Steyn J, Baresi L, Guinea S (2007) A Fault Taxonomy for Web Service Composition. Di Nitto E, Ripeanu M, eds. Proceedings of the 2007, Workshops on Service-Oriented Computing. (Vienna, Austria), 363-375.

Chifu VR, Pop CB, Salomie I, Dinsoreanu M (2010) Web Service Composition Technique Based on a Service Graph and Particle Swarm Optimization. Letia IA, ed. Proceedings of the 2010, International Conference on Intelligent Computer Communication and Processing. (Cluj-Napoca, Romania), 265–272.

Cui L, Kumara S, Lee D (2011) Scenario Analysis of Web Service Composition Based on Multi-criteria Mathematical Goal Programming. Service Science 3(4):280–303.

Dai Y, Yang L, Zhang B (2009) QoS-driven Self-healing Web Service Composition Based on Performance Prediction. Journal of Computer Science and Technology 24(2):250-261.

DocuSign (2014) Using the Payment Processing Feature, https://www.docusign.com/partner/paypal, accessed January 15, 2015.

Erl T (2009) SOA: Principles of Service Design, 5th ed. (Prentice Hall, Upper Saddle River, NJ).

Fakhfakh N, Verjus H, Pourraz F, Moreaux P (2012) QoS Aggregation for Service Orchestrations Based on Workflow Pattern Rules and MCDM Method: Evaluation at Design Time and Runtime. Service Oriented Computing and Applications 7(1):15–31.

Forbes (2010) Global 2000, http://www.forbes.com/lists/2010/18/global-2000-10_The-Global-2000_Rank.html, accessed January 15, 2015.

Forrester (2010) Adoption Of SOA: Still Strong, Even In Hard Times, www.forrester.com/Adoption+Of+SOA+Still+Strong+Even+In+Hard+Times/fulltext/-/E-RES56874?docid=56874, accessed January 15, 2015.

Gao Y, Na J, Zhang B, Yang L, Gong Q, Dai Y (2006) Immune Algorithm for Selecting Optimum Services in Web Services Composition. Wuhan University Journal of Natural Sciences 11(1):221–225.

Grossmann G, Thiagarajan R, Schrefl M, Stumptner M (2011) Conceptual Modeling Approaches for Dynamic Web Service Composition. Kaschek R, Delcambre L, eds. The Evolution of Conceptual Modeling (Springer Berlin Heidelberg), 180–204.

Guo H, Huai J, Li H, Deng T, Li Y, Du Zongxia (2007) ANGEL: Optimal Cconfiguration for High Available Service Composition. IEEE Computer Society, ed. Proceedings of the 2007, IEEE International Conference on Web Services., 280–287.

Huang AF, Lan C, Yang SJ (2009) An Optimal QoS-based Web Service Selection Scheme. Information Sciences 179(19):3309–3322.

Hwang C, Yoon K (1981) Multiple Criteria Decision Making. Lecture Notes in Economics and Mathematical Systems.

Hwang S, Lim E, Lee C, Chen C (2008) Dynamic Web Service Selection for Reliable Web Service Composition. IEEE Transactions on Services Computing 1(2):104–116.

Hwang S, Wang H, Tang J, Srivastava J (2007) A Probabilistic Approach to Modeling and Estimating the QoS of Web-services-based Workflows. Information Sciences 177(23):5484–5503.

Informatica (2014) Global Address Data Quality, http://www.informatica.com/us/products/data-quality/addressdoctor/#fbid=jahmtPV_H-b, accessed January 15, 2015.

Jaeger MC, Muehl G (2007) QoS-based Selection of Services: The Implementation of a Genetic Algorithm. ITG/GI, ed. Proceedings of the 2007, Conference on Communication in Distributed Systems. (Bern, Switzerland), 1–12.

Kieninger A, Berghoff F, Fromm H, Satzger G (2013) Simulation-based Quantification of Business Impacts Caused by Service Incidents. Fãlcão e Cunha J, Snene M, Novoa H, eds. Proceedings of the 2013, International Conference on Exploring Services Science. (Porto, Portugal), 170-185.

Kritikos K, Pernici B, Plebani P, Cappiello C, Comuzzi M, Benbernou S, Brandic I, Kertész A, Parkin M, Carro M (2014) A Survey on Service Quality Description. ACM Computing Surveys 46(1):1–64.

Li J, Ma D, Mei X, Sun H, Zheng Z (2011) Adaptive QoS-aware Service Process Reconfiguration. IEEE Computer Society, ed. Proceedings of the 2011, IEEE International Conference on Services Computing. (Washington, DC), 282–289.

Li J, Zhao Y, Liu M, Sun H, Ma D (2010) An Adaptive Heuristic Approach for Distributed QoS-based Service Composition. IEEE Computer Society, ed. Proceedings of the 2010, International Symposium on Computers and Communicatios. (Riccione, Italy), 687–694.

Li W, Yan-xiang H (2012) A Web Service Composition Algorithm Based on Global QoS Optimizing with MOCACO. Liangzhong J, ed. Proceedings of the 2011, International Conference on Informatics, Cybernetics, and Computer Engineering. (Melbourne, Australia), 79-86.

Lin K, Zhang J, Zhai Y, Xu B (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.

Luo Y, Qi Y, Hou D, Shen L, Chen Y, Zhong X (2011) A Novel Heuristic Algorithm for QoS-aware end-to-end Service Composition. Computer Communications 34(9):1137–1144.

Luthria H, Rabhi F (2009) Using Service Oriented Computing for Competitive Advantage. Kendall KE, Varshney U, eds. Proceedings of the 2009, Americas Conference on Information Systems. (San Francisco, California).

Mani A, Nagarajan A (2002) Understanding Quality of Service for Web Services, http://www.ibm.com/developerworks/webservices/library/ws-quality/index.html, accessed January 15, 2015.

Maolin T, Ai L (2010) A Hybrid Genetic Algorithm for the Optimal Constrained Web Service Selection Problem in Web Service Composition. IEEE Computer Society, ed. Proceedings of the 2010, IEEE World Congress on Computational Intelligence. (Barcelona, Spain), 268–275.

Martello S, Toth P (1987) Algorithms for Knapsack Problems. Annals of Discrete Mathematics 31:213–258.

Maximilien E, Singh MP (2004) A Framework and Ontology for Dynamic Web Services Selection. IEEE Internet Computing 8(5):84–93.

Medjahed B, Malik Z, Benbernou S (2014) On the Composability of Semantic Web Services. Bouguettaya A, Sheng QZ, Daniel F, eds. Web Service Foundations (Springer, New York), 137–161.

Mei L, Chan W, Tse T (2008) An Adaptive Service Selection Approach to Service Composition. IEEE Computer Society, ed. Proceedings of the 2008, IEEE International Conference on Web Services. (Beijing, China), 70–77.

Menascé DA, Casalicchio E, Dubey V (2008) A Heuristic Approach to Optimal Service Selection in Service Oriented Architectures. Avritzer A, Weyuker EJ, Woodside MC, eds. Proceedings of the 2008, ACM International Workshop on Software and Performance. (Princeton, New Jersey), 13–24.

Moghadda M, Davis JG (2014) Service Selection in Web Service Composition: A Comperative Review of Existing Approaches. Bouguettaya A, Sheng QZ, Daniel F, eds. Web Service Foundations (Springer, New York), 321–347.

Ren M, Lyytinen KJ (2008) Building Enterprise Architecture Agility and Sustenance with SOA. Communication of the Association for Information Systems 22(1).

Serviceobjects (2014) Performance Report, http://www.serviceobjects.com/support/performance-and-up-time-reports, accessed January 15, 2015.

Stein S, Payne TR, Jennings NR (2009) Flexible Provisioning of Web Service Workflows. ACM Transactions on Internet Technology 9(1):1–45.

Wan C, Ullrich C, Chen L, Huang R, Luo J, Shi Z (2008) On Solving QoS-Aware Service Selection Problem with Service Composition. IEEE Computer Society, ed. Proceedings of the 2008, IEEE International Conference on Grid and Cooperative Computing. (Shenzhen, China), 467–474.

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

Weise T, Blake BM, Bleul S (2014) Semantic Web Service Composition: The Web Service Challlenge Perspective. Bouguettaya A, Sheng QZ, Daniel F, eds. Web Service Foundations (Springer, New York), 161–189.

Yang L, Dai Y, Zhang B (2009) Performance Prediction Based EX-QoS Driven Approach for Adaptive Service Composition. Journal of Information Science and Engineering (25):345–362.

Yang Z, Shang C, Liu Q, Zhao C (2010) A Dynamic Web Services Composition Algorithm Based on the Combination of Ant Colony Algorithm and Genetic Algorithm. Journal of Computation Information Systems 6(8):2617–2622.

Yu T, Lin KJ (2005) Adaptive Algorithms for Finding Replacement Services in Autonomic Distributed Business Processes. IEEE Computer Society, ed. Proceedings of the 2005, Conference on Autonomous Decentralized Systems. (Sichuan, China), 427–434.

Yu T, Zhang Y, Lin K (2007) Efficient Algorithms for Web Services Selection with end-to-end QoS Constraints. ACM Transaction on the Web 1(1).

Zeng L, Benatallah B, Ngu AHH, Dumas M, Kalagnanam J, Chang H (2004) QoS-aware Middleware for Web Services Composition. IEEE Transactions on Software Engineering 30(5):311–327.

Zeng L, Ngu AHH, Benatallah B, Podorozhny R, Lei H (2008) Dynamic Composition and Optimization of Web Services. Distributed and Parallel Databases 24(1-3):45–72.

Zheng K, Xiong H (2012) A Particle Swarm-based Web Service Dynamic Selection. Journal of Information & Computational Science 9(8):2271–2278.

Zheng Z, Lyu MR (2010) An Adaptive QoS-aware Fault Tolerance Strategy for Web Services. Empirical Software Engineering 15(4):323–345.

APPENDIX

Table 4. Notation	
S_i	Service class S_i that includes all services candidates s_{ij} that implement an action *i* (with *i*=1 to *I*) of the considered execution route
s_{ij}	Service candidate s_{ij} (with *i*=1 to *I*, *j*=1 to J_i) of service class S_i. x_{ij}=1 represents that service candidate s_{ij} is selected for service class *j*; otherwise x_{ij}=0
Υ	Execution route Υ that includes all service classes S_i to implement the corresponding actions *i* of the considered process
sc	Service composition $sc \in SC$ (*SC* represents the set of all possible service compositions of the execution route) is a tuple of service candidates, with exactly one service candidate per service class of the considered execution route
p_{ij}	QoS attribute 'availability' p_{ij} which represents the probability that service candidate s_{ij} does not fail
q_{ij}	QoS vector $q_{ij}=[q_{ij}^1, \dots, q_{ij}^N]^T$ for service candidate s_{ij} including *N* values, each for a single QoS attribute *n* (with *n*=1 to *N*) except of the QoS attribute 'availability'
$q_R^n(s_{ij})$	Expected QoS value $q_R^n(s_{ij})$ for attribute *n* and a single service candidate s_{ij} included in the service composition in case the service candidate fails (considering re-planning options 1 to 3)
Q_c	Global (end-to-end) QoS requirements vector $Q_c=[Q_c^1, \dots, Q_c^N]^T$ for a service composition including *N* values, one (Q_c^n) for each QoS attribute (*n*)
$U(s_{ij})$	Utility $U(s_{ij})$ of a single service candidate s_{ij} based on its QoS vector q_{ij}
$U_R(s_{ij})$	Expected utility $U_R(s_{ij})$ of a single service candidate s_{ij} included in the service composition representing the maximum of the expected utilities of the re-planning options 1 to 3 E[...] (in case the service candidate s_{ij} fails)
$\mathrm{E}[q_{ij}^n, q_R^n(s_{ij}), p_{ij}]$	Expected QoS value for service candidate s_{ij} and attribute *n* based on the QoS value q_{ij}^n, the expected QoS value $q_R^n(s_{ij})$, and the availability p_{ij}
$\mathrm{E}[U(s_{ij}), U_R(s_{ij}), p_{ij}]$	Expected utility for service candidate s_{ij} based on the utility $U(s_{ij})$, the expected utility $U_R(s_{ij})$, and the availability p_{ij}
w	User preference w to weight the QoS attributes in the utility function $U(s_{ij})$ (with w_α for QoS attributes that need to be maximized and w_β for QoS attributes that need to be minimized), where $0<w_\alpha, w_\beta<1$ and $\sum_{\alpha=1}^{x} w_\alpha + \sum_{\beta=1}^{y} w_\beta = 1$ holds
Φ^n	Aggregation function Φ^n for QoS attribute *n* in order to aggregate the QoS values q_{ij}^n of all service candidates s_{ij} included in a service composition. This function is needed to determine the end-to-end QoS values of the service composition
$\widetilde{Q_{ij}}$	QoS vector $\widetilde{Q_{ij}} = [\widetilde{Q_{ij}^1}, \dots, \widetilde{Q_{ij}^N}]^T$ for a service candidate s_{ij} (with *i*=1 to *I*, *j*=1 to J_i) including the random variables $\widetilde{Q_{ij}^n}$ for the QoS attributes *n*=1 to *N* (excluding the QoS attribute availability) with probability distribution $F_Q(q_{ij}^n)$
q_{ij}^n	Realization of the random variable $\widetilde{Q_{ij}^n}$ for a single service candidate s_{ij} with respect to the QoS attribute *n*

Table 5. Service candidates and their QoS values (according to Canfora et al. 2008)					
Service class S_i	Service candidate s_{ij}	Function service candidate	Price	Response Time [ms]	Availability
S_1	$s_{1\,1}$	getAdress1	0.02	7,500	0.89
	$s_{1\,2}$	getAdress2	0.22	5,500	0.91
	$s_{1\,3}$	getAdress3	0.8	3,500	0.93
S_2	$s_{2\,1}$	checkFlight1	0.5	8,000	0.91
	$s_{2\,2}$	checkFlight2	1.5	9,000	0.95
	$s_{2\,3}$	checkFlight3	1.5	9,500	0.95
S_3	$s_{3\,1}$	hotelsearch1	0.5	2,500	0.90
	$s_{3\,2}$	hotelsearch2	0.8	2,000	0.92
	$s_{3\,3}$	hotelsearch3	0.2	3,000	0.88
S_4	$s_{4\,1}$	bookingservice1	3.5	10,000	0.81
	$s_{4\,2}$	bookingservice2	2.5	12,000	0.80
	$s_{4\,3}$	bookingservice3	4.8	8,500	0.82
S_5	$s_{5\,1}$	getHotelInfo1	2.6	4,500	0.97
	$s_{5\,2}$	getHotelInfo2	2.7	4,300	0.98
	$s_{5\,3}$	getHotelInfo3	2.1	7,500	0.96
S_6	$s_{6\,1}$	getShuttleTicketPrice1	0.5	5,500	0.92
	$s_{6\,2}$	getShuttleTicketPrice2	0.55	3,200	0.93
	$s_{6\,3}$	getShuttleTicketPrice3	0.4	4,500	0.92
S_7	$s_{7\,1}$	getTaxiPrice1	0.5	4,000	0.95
	$s_{7\,2}$	getTaxiPrice2	0.9	3,000	0.96
	$s_{7\,3}$	getTaxiPrice3	0.5	4,500	0.94
S_8	$s_{8\,1}$	getDistance1	1.5	2,100	0.97
	$s_{8\,2}$	getDistance2	0.2	5,100	0.88
	$s_{8\,3}$	getDistance3	0.3	4,500	0.90
S_9	$s_{9\,1}$	getCarPrice1	4.5	3,500	0.99
	$s_{9\,2}$	getCarPrice2	1.3	9,500	0.94
	$s_{9\,3}$	getCarPrice3	2.5	7,500	0.97
S_{10}	$s_{10\,1}$	getMetroCardPrice1	0.22	4,500	0.90
	$s_{10\,2}$	getMetroCardPrice2	0.15	5,500	0.87
	$s_{10\,3}$	getMetroCardPrice3	0.5	3,000	0.92

- Insert Figure 8. around here –

- Insert Figure 9. around here -

Table 6. Our approach vs. current selection approaches – all execution routes – deterministic QoS values

Execution route	Service composition	Results based on existing selection approaches			Results based on our approach			Rank order	
		Resp. time	Price	Availability	Expected Resp. time	Expected Price	Expected Utility	Existing approaches	Our Approach
1	$s_{1\ 2}$-$s_{2\ 1}$-$s_{3\ 2}$-$s_{5\ 2}$-$s_{6\ 2}$-$s_{8\ 1}$-$s_{9\ 3}$	17,900	10.77	0.67	23,144	12.43	-2.458	1	270
	$s_{1\ 2}$-$s_{2\ 1}$-$s_{3\ 1}$-$s_{5\ 2}$-$s_{6\ 2}$-$s_{8\ 3}$-$s_{9\ 2}$	26,300	6.07	0.57	29,127	8.18	-2.312	674	1
2	$s_{4\ 1}$-$s_{5\ 2}$-$s_{6\ 2}$-$s_{8\ 1}$-$s_{9\ 1}$	19,900	12.75	0.71	23,831	17.34	-3.008	1	116
	$s_{4\ 3}$-$s_{5\ 2}$-$s_{6\ 1}$-$s_{8\ 3}$-$s_{9\ 3}$	24,800	10.80	0.64	27,401	12.78	-2.708	110	1
3	$s_{1\ 2}$-$s_{2\ 1}$-$s_{3\ 2}$-$s_{5\ 2}$-$s_{6\ 2}$-$s_{8\ 1}$-$s_{10\ 3}$	17,400	6.77	0.62	21,458	6.86	-1.790	1	62
	$s_{1\ 2}$-$s_{2\ 1}$-$s_{3\ 1}$-$s_{5\ 2}$-$s_{6\ 3}$-$s_{8\ 1}$-$s_{10\ 3}$	17,400	6.32	0.60	21,539	6.32	-1.737	8	1
4	$s_{4\ 1}$-$s_{5\ 2}$-$s_{6\ 2}$-$s_{8\ 1}$-$s_{10\ 3}$	19,400	8.75	0.66	22,333	8.84	-2.042	1	10
	$s_{4\ 1}$-$s_{5\ 1}$-$s_{6\ 1}$-$s_{8\ 3}$-$s_{10\ 3}$	22,000	7.40	0.60	24,429	7.52	-2.008	54	1
5	$s_{1\ 2}$-$s_{2\ 1}$-$s_{3\ 2}$-$s_{5\ 2}$-$s_{7\ 1}$-$s_{8\ 1}$-$s_{9\ 1}$	17,900	10.72	0.68	22,938	12.45	-2.450	1	346
	$s_{1\ 2}$-$s_{2\ 1}$-$s_{3\ 3}$-$s_{5\ 2}$-$s_{7\ 1}$-$s_{8\ 3}$-$s_{9\ 3}$	24,800	6.92	0.59	28,257	8.58	-2.311	642	1
6	$s_{4\ 1}$-$s_{5\ 2}$-$s_{7\ 1}$-$s_{8\ 1}$-$s_{9\ 1}$	19,900	12.70	0.72	23,857	18.05	-3.084	1	145
	$s_{4\ 3}$-$s_{5\ 1}$-$s_{7\ 1}$-$s_{8\ 1}$-$s_{9\ 2}$	24,600	10.70	0.69	26,954	13.14	-2.723	49	1
7	$s_{1\ 2}$-$s_{2\ 1}$-$s_{3\ 2}$-$s_{5\ 2}$-$s_{7\ 1}$-$s_{8\ 1}$-$s_{10\ 3}$	17,400	6.72	0.63	21,452	6.80	-1.783	1	28
	$s_{1\ 2}$-$s_{2\ 1}$-$s_{3\ 1}$-$s_{5\ 2}$-$s_{7\ 1}$-$s_{8\ 3}$-$s_{10\ 3}$	19,800	5.22	0.57	23,291	5.61	-1.751	197	1
8	$s_{4\ 1}$-$s_{5\ 2}$-$s_{7\ 1}$-$s_{8\ 1}$-$s_{10\ 3}$	19,400	8.70	0.67	22,311	8.96	-2.053	1	12
	$s_{4\ 1}$-$s_{5\ 2}$-$s_{7\ 1}$-$s_{8\ 3}$-$s_{10\ 3}$	21,800	7.50	0.62	24,141	7.70	-2.013	37	1

Table 7. Results of our approach with vs. without considering non-deterministic QoS values – all execution routes

Execution route	Service composition	Results based on deterministic QoS values			Results based on non-deterministic QoS values			Rank order	
		Expected Resp. time	Expected Price	Expected Utility	Expected Resp. time	Expected Price	Expected Utility	Deter. QoS	Non-Deter. Qos
1	$s_{1\ 2}$-$s_{2\ 1}$-$s_{3\ 1}$-$s_{5\ 2}$-$s_{6\ 2}$-$s_{8\ 3}$-$s_{9\ 2}$	29,127	8.18	-2.312	29,342	9.82	-2.485	1	27
	$s_{1\ 2}$-$s_{2\ 1}$-$s_{3\ 3}$-$s_{5\ 2}$-$s_{6\ 1}$-$s_{8\ 2}$-$s_{9\ 3}$	28,933	9.11	-2.386	29,109	9.40	-2.440	114	1
2	$s_{4\ 3}$-$s_{5\ 2}$-$s_{6\ 1}$-$s_{8\ 3}$-$s_{9\ 3}$	27,401	12.78	-2.708	27,526	14.48	-2.883	1	45
	$s_{4\ 3}$-$s_{5\ 2}$-$s_{6\ 1}$-$s_{8\ 3}$-$s_{9\ 1}$	24,679	14.24	-2.714	24,750	14.81	-2.781	3	1
3	$s_{1\ 2}$-$s_{2\ 1}$-$s_{3\ 1}$-$s_{5\ 2}$-$s_{6\ 3}$-$s_{8\ 1}$-$s_{10\ 3}$	21,539	6.32	-1.737	21,964	6.68	-1.791	1	16
	$s_{1\ 2}$-$s_{2\ 1}$-$s_{3\ 1}$-$s_{5\ 1}$-$s_{6\ 2}$-$s_{8\ 1}$-$s_{10\ 3}$	21,791	6.45	-1.758	21,994	6.31	-1.759	8	1
4	$s_{4\ 1}$-$s_{5\ 1}$-$s_{6\ 1}$-$s_{8\ 3}$-$s_{10\ 3}$	24,429	7.52	-2.008	24,500	8.27	-2.072	1	10
	$s_{4\ 1}$-$s_{5\ 2}$-$s_{6\ 3}$-$s_{8\ 3}$-$s_{10\ 3}$	24,197	7.84	-2.026	24,265	7.74	-2.023	5	1
5	$s_{1\ 2}$-$s_{2\ 1}$-$s_{3\ 3}$-$s_{5\ 2}$-$s_{7\ 1}$-$s_{8\ 3}$-$s_{9\ 3}$	28,257	8.58	-2.311	28,483	10.20	-2.490	1	642
	$s_{1\ 2}$-$s_{2\ 1}$-$s_{3\ 3}$-$s_{5\ 2}$-$s_{7\ 3}$-$s_{8\ 2}$-$s_{9\ 2}$	30,077	8.43	-2.384	30,179	9.10	-2.451	92	1
6	$s_{4\ 3}$-$s_{5\ 1}$-$s_{7\ 1}$-$s_{8\ 1}$-$s_{9\ 2}$	26,954	13.14	-2.723	26,948	13.92	-2.799	1	2
	$s_{4\ 3}$-$s_{5\ 2}$-$s_{7\ 3}$-$s_{8\ 1}$-$s_{9\ 1}$	22,495	16.37	-2.839	22,319	15.87	-2.778	51	1
7	$s_{1\ 2}$-$s_{2\ 1}$-$s_{3\ 1}$-$s_{5\ 2}$-$s_{7\ 1}$-$s_{8\ 3}$-$s_{10\ 3}$	23,291	5.61	-1.751	23,629	5.87	-1.794	1	5
	$s_{1\ 2}$-$s_{2\ 1}$-$s_{3\ 3}$-$s_{5\ 2}$-$s_{7\ 3}$-$s_{8\ 3}$-$s_{10\ 3}$	23,917	5.54	-1.773	24,065	5.48	-1.776	13	1
8	$s_{4\ 1}$-$s_{5\ 2}$-$s_{7\ 1}$-$s_{8\ 3}$-$s_{10\ 3}$	24,141	7.70	-2.013	24,174	8.32	-2.077	1	11
	$s_{4\ 3}$-$s_{5\ 2}$-$s_{7\ 1}$-$s_{8\ 3}$-$s_{10\ 3}$	22,681	8.89	-2.062	22,734	8.58	-2.034	23	1

Figure 1. *Travel booking process*

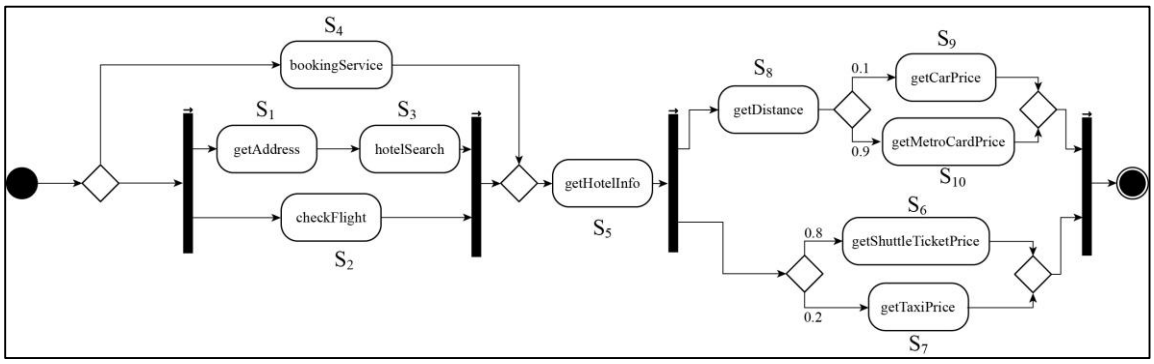

Figure 2. *Simplified excerpt of the process of the running example*

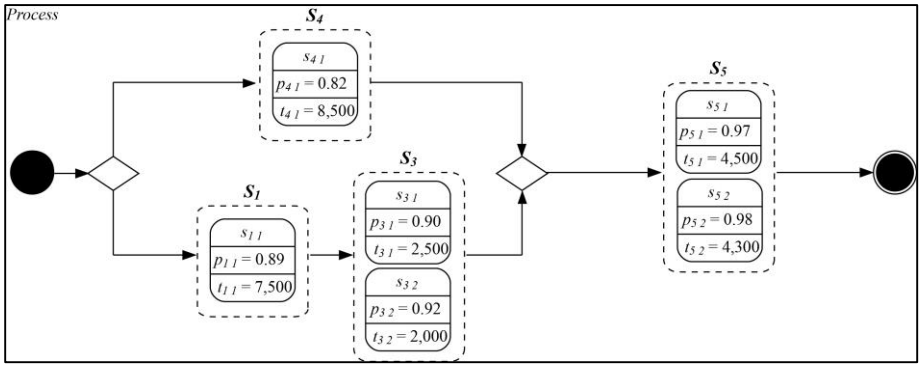

Figure 3. *Calculating the expected utility for re-planning option 1*

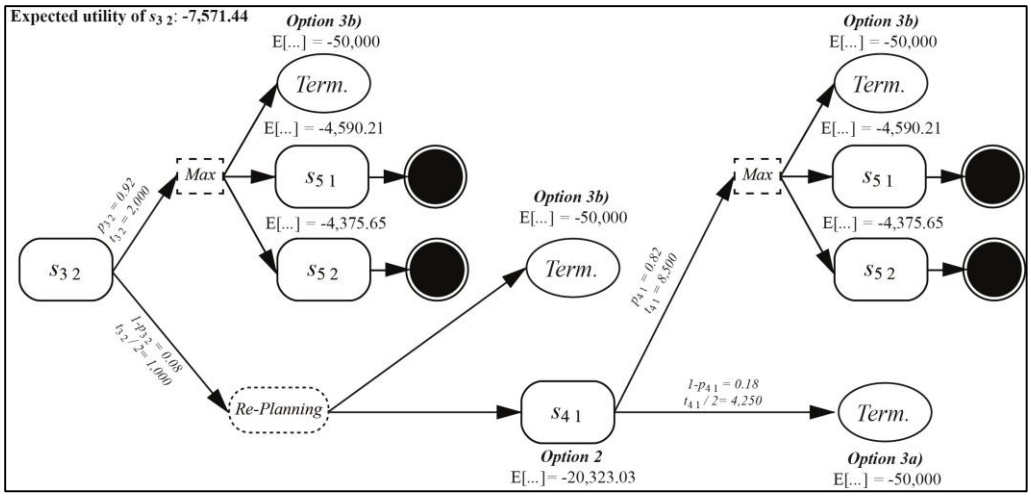

Figure 4. *Calculating the expected utility for re-planning option 2*

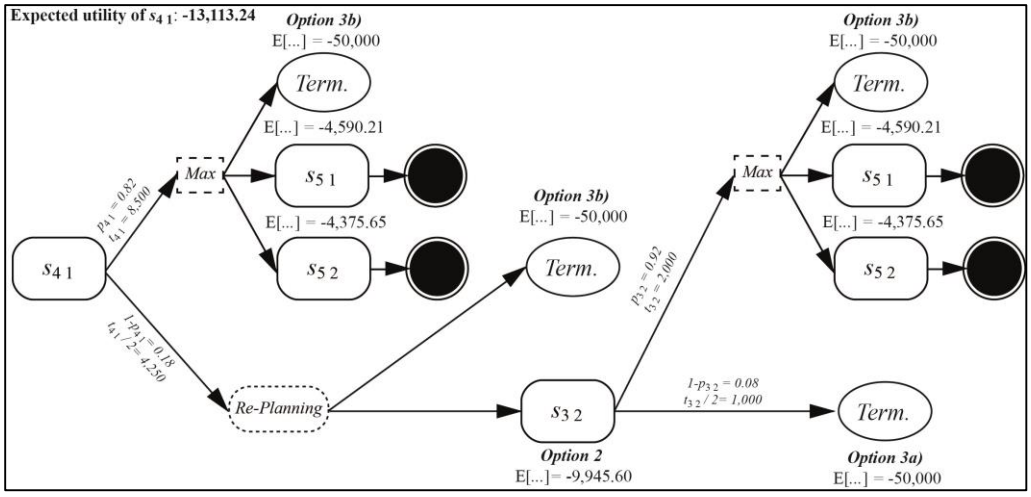

Figure 5. *Histogram response time DOTS address validation*

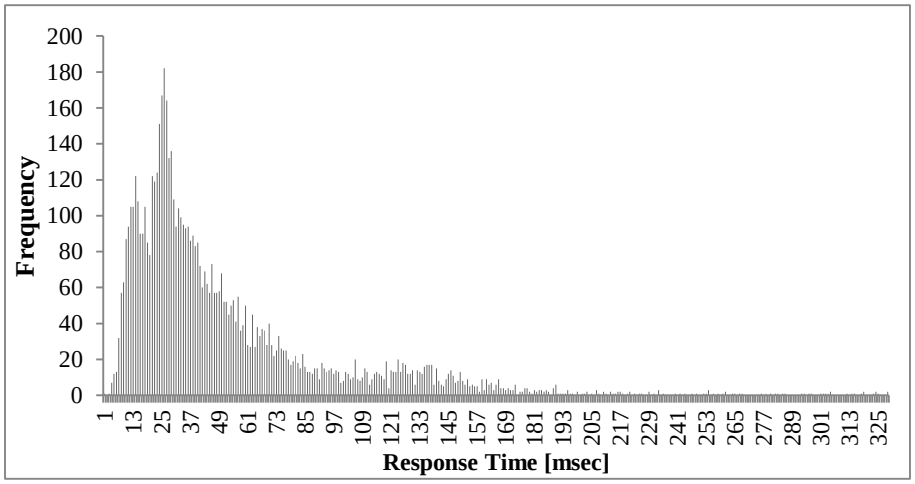

Figure 6. *Excerpt of the example*

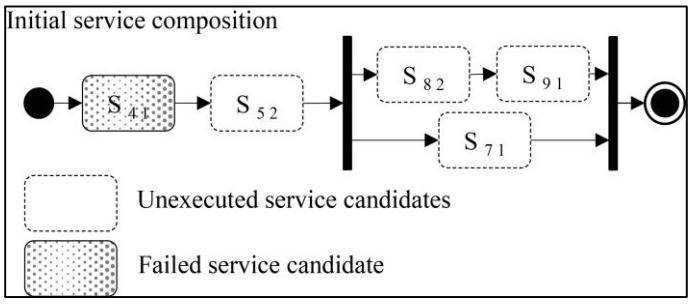

Figure 7. *Performance of solving the simulation model*

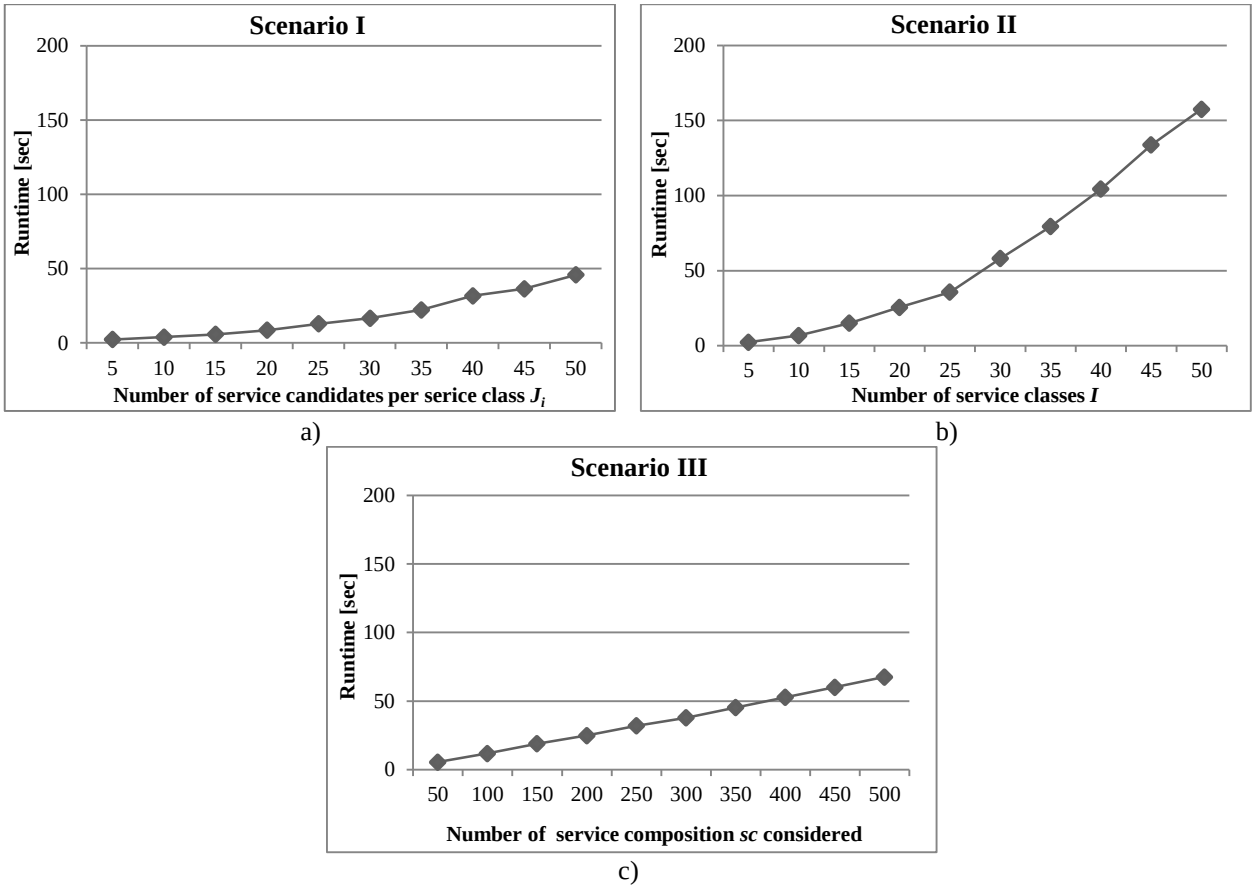

Figure 8. *Simulation procedure*

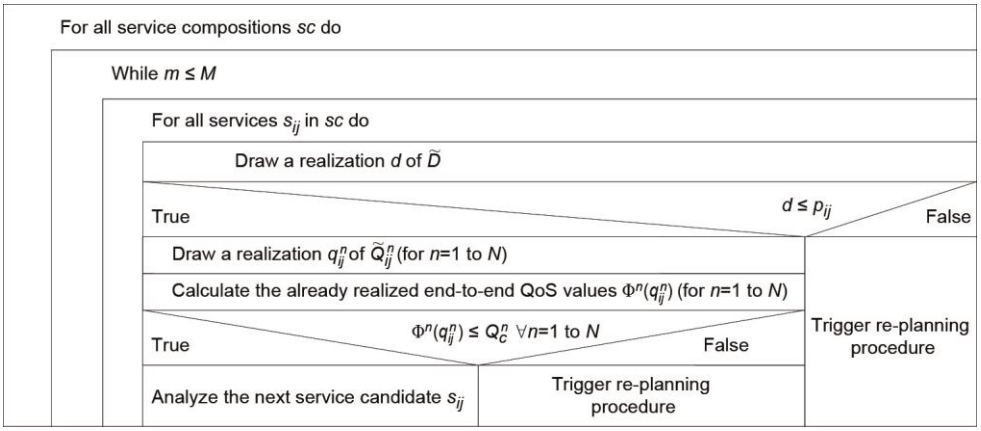

Figure 9. *Re-planning procedure*