One Size Does Not Fit All: A Group-based Service Selection for Web-based Business Processes

Basem Suleiman*, Carlos Eduardo da Silva† and Sherif Sakr‡

*NICTA, UNSW and SAP Research, Sydney, Australia
Email: Basem.Suleiman@nicta.com.au
†School of Computing, University of Kent, Canterbury, Kent, UK
Email: ces26@kent.ac.uk
‡National ICT Australia(NICTA) and University of New South Wales, Sydney, Australia
Email: ssakr@cse.unsw.edu.au

Abstract—Web service failures or degradations directly cause operational inefficiencies and financial losses in Web service-based business processes (WSBP). In today’s competitive Web service market a considerable number of functionally-similar Web services offered by different providers are differentiated by competitive QoS levels and pricing structures. Consequently, dynamic and optimal Web service selection is a significant challenge to business organizations which run such WSBPs. Having a cost-effective and efficient Web service selection approach is becoming an important necessity for such organizations. Current Web service selection approaches offer “one-size-fits-all” solution, i.e., one optimal service selection for all running BP instances. However, such solutions are neither efficient nor cost-effective given that the service levels of WSBPs are associated with customers classes/profiles, e.g., Gold, Silver or Bronze. Therefore, we propose a group-based Web service selection approach, “one-size-does-not-fit-all”, that optimizes multiple QoS criteria and differentiates the service selection based on the BP customers classes/profiles. Our approach shows a very good improvement over existing “one-size-fits-all” approaches; 65% average cost reduction and 30% utility value improvement.

I. INTRODUCTION

In today’s competitive and fast growing business climate, most of organizations’ operations and services are realized as automated business processes (BPs). These processes are crucial as they aim to deliver business value for an organization as well as its customers. The underlying technology, specifically Web services, which enables realizing and managing these BPs has rapidly advanced. In essence, the Web services paradigm has become prominent because it facilitates the distributed implementation of BPs.

A BP is comprised of a set of activities that achieve a specific business service or goal. These activities could be: (a) human tasks that require manual input or processing of data; (b) automated activities that either access internal back-end system or enterprise services, i.e., within corporate walls, or external software system, i.e., Web services or (c) hybrid activities, i.e., combination of both human and automated system tasks. In this paper we use the term Web service-based business processes (WSBP) to refer to BPs of type (b) and (c). Examples of such BPs can be found in enterprise systems such as supply chain management (SCM) and customer relationship management (CRM).

The realization of some BP activities as Web service implementation has some benefits such as process agility and improvement of organization’s competitive advantage. However, it also has significant drawbacks. Web services are highly dynamic in nature which means that the service performance and its quality behaviour are to some extent unpredictable [1]. This behaviour has a direct impact on delivering quality business services to the BP consumers and therefore influencing business process performance measures [2]. Often, such impact automatically translates into financial losses and customer frustration. In the cost of downtime online survey [3] it was reported that 46% of those participant companies said each hour of downtime would cost their companies up to $50K.

Although there could be several sources of service failure or degradation (e.g., hardware/software, errors), its negative effects are similar, e.g., financial losses and customer frustration. To reduce the impact of such negative effects, it is crucial to take some immediate actions particularly when one or more service fails or degrades. One way of dealing with such situation is through the adaptation of the BP, e.g., replacing the faulty or under-performing Web service at runtime. In this paper, we address the problem of how to dynamically select Web service offerings from different service providers to replace faulty or degraded Web services that implement automated activities in a BP. The selection is mainly based on multiple BP metrics (both technical and business) which are mapped into relevant QoS properties offered by available Web services. It focuses on optimizing multiple criteria, i.e., BP metrics, at the level of BP instances.

While there are large number of Web service selection and optimization approaches in the body of the literature [4], [5], [6], [7], these approaches propose “one-size-fits-all” solutions which assume that there is an ideal service selection that suits different running WSBP instances (or different
selection criteria for each running instance). However, in practice, this is not the case, as there may be different WSBP instances associated with different classes of customers. For example, organizations nowadays offer their consumers different classes of services based on criteria such as customer type or profile and agreed service levels [8], [9]. We argue that Web service selection decisions should be differentiated based on certain criteria that contribute to an organizations’ business objectives and strategy [2] (e.g. considering customers’ profiles during the selection process). In this paper we propose a group-based service selection; “one-size-does-not-fit-all” approach and investigate its potential positive impacts on the utility maximization and cost minimization.

Given the characteristics of the service selection problem described above, we adopt Constraint Programming (CP) as an approach to address this challenge. CP has proven its ability to effectively and efficiently find an optimal solution for combinatorial problems [10]. We also utilize multi-criteria optimization to achieve optimal service selection that optimizes multiple criteria based on defined objective functions. This is important as in practice the number of service offerings have been growing 1, 2 and these offerings are very competitive in terms of the number of business and QoS metrics. Furthermore, usually there is a large number of QoS and business criteria (represented as constraints) that need to be optimized during the service selection process which make service selection a more complex problem. There are several solver algorithms and techniques for solving and optimizing a solution search in constraint satisfaction problems. Such tools (e.g., MS solver foundation 4, IBM constraint solvers 5 and others 6) are based on systematic search algorithms and Artificial Intelligence techniques, which can be utilized for modelling and solving the service selection problem.

The contribution of this paper is an group-based Web service selection approach that optimizes service selection based on certain business and QoS criteria and objectives which correspond to different BP instances grouped based on customer classes and their requirements. It also considers the selection of different service offerings provided by different service providers to satisfy business and QoS requirements of different groups of BP instances which share similar properties. Compared to traditional service selection approaches, our group-based selection neither adheres to one service offering nor adheres to offerings of one service provider to satisfy the requirements of all BP instances. Instead it selects the best offers from different providers to meet requirements of different groups of instances so that cost is minimized and utility is maximized.

This paper is organized as follows, section II presents an example scenario used for explaining our approach. Section III describes our approach, presenting a formal model of the group-based service selection problem; experimental results are presented in section IV, while section V presents some related work. The conclusions and future work are presented in section VI.

II. Motivating Scenario: A Home Loan WSBP

Consider a online home loan BP provided by MaxSavings financial organization (shown in Figure 1). The process contains some human-task activities such as “Collect Personal Information” as well as system-tasks such as “Obtain Customer Profile”. The system-tasks are either implemented as back-end systems, e.g., CRM, ERP, or outsourced to external specialized service providers, e.g., “Check Credit History” and “Property Valuation”.

MaxSavings also differentiates the level of service provided to its BP customers based on their profiles, i.e., customers are classified as Gold, Silver and Bronze classes according to their financial investments and types. Imagine that 200 customers, of different classes, are lodging their home loan application the “Check Credit History” Web service fails to respond or its service level degrades. MaxSavings has some business performance metrics derived from the corporate strategy and goals. Due to its significant impact on delivering desired quality and high value services [2] to its BP consumers, MaxSavings’ management enforces these metrics and their mapping at the operational level, both business and IT. It has different business-IT metrics which estimate incurred costs or losses that could result from software problems, e.g., Web services failures/degradations. Consequently, it is crucial to consider the impact of any change in the services layer on the performance measures at the process layer. Furthermore, an important part of MaxSavings’s strategy is to ensure consistent and high service levels for its customers based on their classes. For example, reducing the waiting time at each process activity is essential as it reduces the possibility of performance bottleneck and burden on hardware resources and human support costs as customers are unable to complete their home loan application when a Web service fails or degrades. The service levels differ based on the BP customer profiles and classes; for instance the service response time to a Gold class customer is lower than the one to Silver class customer.

These are some key motivations behind the need to dynamically adapt BP, particularly replacing a faulty/under-performing Web service that satisfies multiple objectives or criteria derived from business-IT metrics and based on different criteria associated with the user class. For MaxSavings, it is important to select the most cost-effective and the best possible QoS values or combination of them, that leads to satisfactory process measures. Given different classes of process customers (e.g., Gold, Silver, Bronze), we argue that

---

1 http://webservices.seekda.com/
2 http://demo.service-finder.eu/search
4 http://code.msdn.microsoft.com/solverfoundation
6 http://wwwconstraint-solving.com/solvers

254
the selection decision should be differentiated depending on the customer class to which the instances belong.

III. ONE SIZE DOES NOT FIT ALL APPROACH

In this section we introduce our group-based service selection approach. As shown in Figure 2, the overall approach consists of several steps. First the service selection problem (SSP) is formally modelled as a multi-criteria optimization problem (MCOP). The formalism of our group-based service selection approach is explained in detail in section A. A user interface could be developed in order to enable process designers to easily specify the selection variables, domains, constraints and objective functions. The resulting set of files from this step are then stored in a repository which contains all formal service selection models. The models which belong to different BPs can be used as an input for a constraint solver tool. We assume a transformation plug-in can be used to transform our formal model into the solver tool language. This enhances maintainability and enables the use of different solvers independently or together. The service selection files are then fed into a constraint solver tool which can run appropriate solver algorithms over the models to find the optimal service selection. There are several constraint solver tools that can be used to solve MCOP. We use Microsoft solver foundation\(^7\) as it has several advantages. First, different languages including MS optimization Modelling Language (OML), C\(^\sharp\), F\(^\sharp\), or any .NET Framework language can be used to model optimization problems. Second, several integrated third-party solvers can also be used such as Gurobi, Mosek and CPLEX. Furthermore, other solvers can also be added as an extension. Third, modelling and solving models can be done using familiar interfaces such as MS Excel.

\(^7\)http://code.msdn.microsoft.com/solverfoundation

A. Modelling Group-based Service Selection

The selection of a Web service offering is modelled as a multi-criteria constraint optimization problem. In the following definitions, we present a formal modelling of our group-based service selection problem.

In our approach, we are considering the perspective of the business process owner by defining a process \(P\) which has a number of classes differentiated based on the process quality requirements. These classes of a business process \(P\) are defined by the following set:

\[
P = \{p_1, p_2, \ldots, p_k\},
\]

where \(k\) is the number of BP classes, which in this work is fixed to 3 (e.g., \(P = \{p_{\text{Gold}}, p_{\text{Silver}}, p_{\text{Bronze}}\}\)).

Each business process \(P\) has different automated activities that are realized by Web services of different functionality. These Web service functionalities are defined by the following set:

\[
SF = \{S_{f_1}, S_{f_2}, \ldots, S_{f_m}\},
\]

where \(m\) represent the number of Web services used in the business process \(P\) (e.g., in our MaxSavings home loan BP \(SF = \{\text{Check Credit History, Property Valuation, Area demographic}\}\)). It is important to mention that in our approach, we are considering the selection of a replacement for one Web service functionality at a time, e.g., selecting one Web service replacement for under-performing/failed service “Property Valuation” in the home loan BP; and the selection of a Web service offering for each process class will happen independently, by considering the requirements associated with that particular class. In this way, whenever the selection process is started, it is for a specific service functionality.

A Web service functionality is realized by a Web service offering provided by a service provider. In this context, the
Figure 2. Overview of the service selection approach.

Different Web service providers are defined by the set:

\[ S = \{ S_1, S_2, ..., S_i \} \]

where \( i \) corresponds to the number of available Web service providers. For example, the following is the set of candidate “Property Valuation” service functionality offered by three service providers:

\[ S = \{ \text{PropertyValuation}_1, \text{PropertyValuation}_2, \text{PropertyValuation}_3 \} \]

It is also the case that a Web service provider \( i \) may differentiate its Web service offerings of the same functionality based on quality properties and pricing schemes. We refer to each of these service offerings as a Web service instance. Based on this, the candidate Web service instances (functionally-similar services) offered by different Web service providers are defined by the following set:

\[ S_{ij} = \{ S_{i1}, S_{i2}, ..., S_{ij} \} \]

where \( i \) identifies the service provider, and \( j \) is the number of service instances that belong to the \( i^{th} \) provider.

Each candidate Web service provider \( i \) can offer a different number of service instances with different quality attributes and pricing schemes for a given functionality, and the number of classes considered by the BP owner is independent of the number of service instances provided by third part service providers. In this context, the number of service instances of different service providers can also be different, i.e., \( j \) is not equal for all service providers. As an example, two different service providers which offer, respectively, two and three service instances for the Property Valuation functionality can be denoted as \( S_{1j} = \{ \text{PropertyValuation}_{11}, \text{PropertyValuation}_{12} \} \) and \( S_{2j} = \{ \text{PropertyValuation}_{21}, \text{PropertyValuation}_{22}, \text{PropertyValuation}_{23} \} \)

As previously mentioned, our group-based selection process considers the selection of a Web service instance for a particular service functionality. Since we are assuming that all candidate Web service instances that provide a particular functionality have the same set of QoS properties therefore, the set of QoS properties associated with a particular service functionality \( S_{fm} \) is captured by the following set:

\[ S_{fm}.Q = \{ S_{fm}.q_1, S_{fm}.q_2, ..., S_{fm}.q_n \} \]

where \( n \) is the number of properties considered, which in our case is fixed and equal to eight QoS properties for all candidate Web service instances offered by all service providers. We assume these QoS properties have the same semantic meaning and metrics. We use OASIS Web Service Quality Model (WSQM) for Web Services\(^8\) for specifying and interpreting QoS properties. We choose the most important service level management and business value qualities.

The following is an example of the QoS properties associated with the Property Valuation Web service functionality:

\[ S_{f2}.Q = \{ s_{f2}.ResponseTime, s_{f2}.MaxThroughput, s_{f2}.Availability, s_{f2}.Accessability, s_{f2}.Successability, s_{f2}.Price, s_{f2}.ServiceReputation, s_{f2}.ProviderReputation \} \]

Each Web service instance has a set of values for its QoS properties. These values are defined as follows:

\[ S_{ij}.QV = \{ S_{ij}.qv_1, S_{ij}.qv_2, ..., S_{ij}.qv_n \} \]

where \( S_{ij}.qv_n \) represents the value associated with the \( n^{th} \) QoS property of the \( j^{th} \) Web service instance offered by the \( i^{th} \) service provider. Given the above example, the values associated with service instance \( S_{11} \) can be denoted as follows \( S_{11}.Q = \{ 30ms, 150T/sec, 98.4\%, 98\%, 99\%, $5.5, 8.1, 8.5 \} \)

From the BP owner, each Web service functionality, that implements an automated activity, can have different classes of quality, each class is associated with a different business and quality requirements that must be considered in the selection process. These requirements are associated with

\(^8\)http://www.oasis-open.org/committees/documents.php?wg_abbrev=wsqm/
the different QoS properties of the different service type (functionality) of the BP. In this context, we define these requirements (constraints) as the following set:

$$R_k \cdot S_{f_m} \cdot Q = \{ R_k \cdot S_{f_m} \cdot q_1, R_k \cdot S_{f_m} \cdot q_2, \ldots, R_k \cdot S_{f_m} \cdot q_n \},$$

where \( R_k \cdot S_{f_m} \cdot q_n \) corresponds to QoS requirement associated with each property \( n \) of service functionality \( m \) of the \( k \)th business process class. The number of these QoS requirements is assumed to be equal to the number of QoS properties of all service instances offered by all the service providers. These QoS requirements represent the constraints that must be satisfied by the candidate Web service instances during the group-based service selection process.

Considering all the values associated with all service instances that offer a particular functionality, and the requirements associated with this functionality, we can identify the initial solution set \( S_{I_k} \cdot S_{f_m} \) as the set of all Web service instances \( S_{ij} \) that provides a particular functionality \( m \), where \( S_{ij} \cdot q_n \) satisfy all the constraints \( R_k \cdot S_{f_m} \cdot q_n \) for each property \( q_n \). This solution set contains all Web service instances whose quality attributes satisfy all constraints simultaneously. However, it is still necessary to identify the best/optimal Web service instance based on certain given optimization criteria e.g., minimum cost.

To find the optimal/best Web service instance, we apply a multi-criteria optimization technique, where the goal is to maximize an overall utility in terms of an objective function, considering the QoS properties of the Web services. These different QoS properties have different importance weights, based on the BP class. To identify these different priorities, we define for each BP class a weight vector \( PC_k \cdot S_{f_m} \) that identifies the importance value associated with each QoS property of functionality \( S_{f_m} \) for each BP class \( PC_k \). This vector is defined as follows:

$$PC_k \cdot S_{f_m} \cdot W = \{ PC_k \cdot S_{f_m} \cdot w_1, \ldots, PC_k \cdot S_{f_m} \cdot w_n \},$$

where \( n \) is the number of QoS properties associated with functionality \( m \).

The QoS properties considered for identifying the optimal/best service may have distinct types, and value ranges. In order to allow a comparison between the different Web services and the different QoS properties, it is necessary to quantify the different properties in a common mathematical function that can be used for calculating the best overall utility of the Web services in the solution space. In order to do that, a normalization process is applied to all QoS values of all Web service instances before the overall utility value is calculated for each service instance. The normalization process considers the types of the properties, the domain of possible values, and the associated weights for applying a “scoring function” (e.g., a gaussian function) to determine the score associated with each property. This normalization requires the identification of all possible values associated with the QoS properties based on the candidate services. In this context, given all the values associated with all Web service instances of a particular functionality, we can identify the domain of each property \( S_{f_m} \cdot q_n \) as the following set:

$$D \cdot S_{f_m} \cdot Q_n = \{ S_{11} \cdot q_n, ... , S_{i1} \cdot q_n, S_{12} \cdot q_n, ... , S_{ij} \cdot q_n, ... , S_{1n} \cdot q_n \},$$

where \( S_{ij} \cdot q_n \) is the value associated with property \( q_n \) for all Web service instances \( j \) of the service provider \( i \) that provide functionality \( m \). This set identifies all possible values for property \( S_{f_m} \cdot q_n \) based on the available service instances.

Once the domain of each QoS property has been identified, the normalized value for property \( S_{ij} \cdot q_n \) of candidate Web service instance \( S_{ij} \) is calculated by applying the following formula:

$$F(S_{ij} \cdot q_n, norm) = 1 - \frac{MAX(D \cdot S_{f_m} \cdot Q_n) - S_{ij} \cdot q_n}{MAX(D \cdot S_{f_m} \cdot Q_n) - MIN(D \cdot S_{f_m} \cdot Q_n)} \text{ if } PC_k \cdot S_{f_m} \cdot w_n \geq 0$$

$$F(S_{ij} \cdot q_n, norm) = \frac{MAX(D \cdot S_{f_m} \cdot Q_n) - S_{ij} \cdot q_n}{MAX(D \cdot S_{f_m} \cdot Q_n) - MIN(D \cdot S_{f_m} \cdot Q_n)} \text{ if } PC_k \cdot S_{f_m} \cdot w_n < 0$$

where, \( MAX(D \cdot S_{f_m} \cdot Q_n) \) is the maximum QoS value of property \( q_n \) for all Web service instance candidates, and \( MIN(D \cdot S_{f_m} \cdot Q_n) \) is the minimum QoS Value of property \( q_n \) for all Web service instance candidates. Note that these equations are for normalizing QoS values of numeric data types. For other QoS values’ data types, e.g., Boolean and set data types, different equations must be used. In the context of this paper, we only consider numeric data types. Finally, the normalized QoS values (e.g., \( S_{ij} \cdot q_n, norm \)) for property \( q_n \) of service instance \( S_{ij} \) are used for calculating the total utility of each candidate service instance through the following objective function:

$$F(S) = \sum(S_{ij} \cdot q_n, norm \ast PC_k \cdot S_{f_m} \cdot w_n)$$

The best service instance is the one that has the maximum value for this objective function, e.g., maximum utility. There could be one or more objective functions defined for each group of process instances.

**IV. Evaluation**

In order to evaluate our group-based service selection approach, we conducted experiments comparing “one-size-fits-all” solution against our group-based solution (“one-size-does-not-fit-all”). The experiments involved a BP with three different business classes (Gold, Silver, Bronze) where it is necessary to select a service offer for each process instance. We considered two criteria for comparing the different selection mechanisms. The first criteria is the minimization of the total cost while the second criteria considered the maximization the utility associated with the selected offers.

Based on these two criteria, we have compared three selection mechanisms. The first mechanism (M1) represent the situation where a particular service offer is selected to satisfy the request of all process instances. In other words, this mechanism sticks to one service offer that satisfy all business classes. The second selection mechanism (M2)
represents our approach, where we dynamically look for the best combination of service offers from different providers for the instances from the different BP classes.

Table I represents the characteristics of different service offers of online credit card check Web services which used in our experiments. The naming convention\(^9\) adopted in this table is interpreted as follows: \(S_{xy}\) represents the service offer \(y\) from provider \(x\). We randomly generated the values of QoS attributes that are not provided in the providers' offers. Based on these offers, Table II identifies the selected service offer(s) of each selection mechanism for the different classes of process instances for the two criteria considered in this work (cost and utility). Table III presents the different request workloads used in our experiments where each workload (\(W_i\)) consists of a number of process instances from each class (Gold, Silver, Bronze).

Figure 3 presents the results for our experimental comparison between the different selection mechanisms in terms of their financial cost (3(a)) and achieved utility (3(b)). The results show the effectiveness of our approach in that achieves better performance in both criteria (minimum minimum
financial cost and the maximum utility). In particular, in terms of financial cost our approach shows an average reduction of 65% and in terms of the achieved utility, our approach achieves an average improvement of 30%. Figure 3(b) shows the benefits of our approach in terms of the achieved utility, with a clear improvement in the utility of all classes (Gold, Silver and Bronze), mainly the utility associated with the Gold class, which captures the most important process instances (clients). In these experiments, we have considered an equal-weight for the different QoS properties in constituting the utility function. However, these results confirm the advantage of our approach in considering different BP classes when selecting satisfying service offers independently of the assigned weights of the different QoS properties or the used selection criteria.

V. RELATED WORK

The services paradigm provides the infrastructure for enterprises to build their applications by integrating existing individual services on demand. However, it is important for enterprises to be able to select their service providers dynamically based on the characteristics of their customer classes to optimize the metrics of their ultimate goals.

Some Web service selection approaches [5] and [6] were based on multi-criteria decision making (MCDM) methods. Seo et al. [5] based their Web service selection approach on Preference Ranking Organization Method for Enrichment Evaluations (PROMETHEE) method. They used six criteria functions for calculating selection criteria preferences, the difference between two alternatives for any criteria. The multi-criteria preference index was used to compute the weighted average of the preference functions for all criteria. In addition, the alternatives’ strengths/weaknesses were calculated using "outranking flows". Meanwhile, Tran et al. [6] introduced a QoS-based ranking algorithm which was based on Analytical Hierarchy Process (AHP) method along with a comprehensive QoS ontology, WS-QoSOnto. They also introduced rules for evaluating relative ranking between two services based on QoS properties of various value types and units. The ranking algorithm itself includes four steps; constructing AHP hierarchy for QoS ranking, computing relative weighting and ranking of QoS properties and iterative accumulation of all ranking calculations to select the service with the highest rank. Both approaches did not consider the performance overhead they add. Particularly, the exhaustive pair-wise QoS comparisons, ranking and weighting calculations required by PROMETHEE and AHP are tedious and time consuming when there is large number of candidate services and/or QoS properties as the number of QoS comparisons will grow exponentially. Both approaches [5],[6] evaluated on small data set size; for example, in [5] few selection criteria and candidate services were used and in [6] few service providers were included. Therefore, the scalability of both approaches will be an important concern. This is one advantage of using an MCOP method because it which uses intelligent algorithms to reduce the search space by iteratively testing the satisfaction of predefined constraints.

Dubey and Menasce [7] investigated the effectiveness of an extended JOSeS algorithm and a heuristic algorithm to find optimal and near-optimal service selections respectively. The optimization was based on maximizing a utility function of multiple QoS and cost constraints. The two selection algorithms were compared based on the number of points examined in the solution space, computation time, the tightness of QoS and cost constraints, the number of service candidates. While the experiments showed useful conclusions regarding both algorithms’ effectiveness with different settings, the selection algorithms did not consider the effect of group-based selection and variability of service providers’ offerings on optimizing QoS and cost criteria. Our group-based approach shows an optimal service selection, utility maximization of multiple QoS criteria, and cost minimization, can be achieved when considering group-based service selection based on customer classes and multiple provider offerings of variable QoS and cost values.

Some other approaches [11], [12], [13], [14] consider the classification of users according to their preference/profiles, context and usage pattern in their optimal service selection approaches. Our approach has essential distinctions; 1) the selection is customized at a fine-granular level where process instances are grouped based on similar QoS and cost constraints that are significant to the organization who own the BP, 2) It considers not only different user classes grouped by QoS criteria but also various service offerings offered by the same and different service providers, 3) It dynamically combines different service selections from different providers offerings to achieve maximum utility and minimum costs. In other words, it does not necessarily stick to one provider offers for all groups of BP instances.

Several research approaches including [15], [16], [17], [18], [19] attempted to address the service selection problem in the context of optimal service composition that satisfies certain objectives, e.g., overall execution time and cost. In such approaches each service selection depends on other service selections in the composition where trade-offs between different service selections can be made to reach overall optimal composition. For example, if some selection criteria of one service selection are not met at a certain stage in the composition, other service selections could balance the difference in the overall optimization functions, e.g., total response time. Such approaches are not appropriate when the service selection is performed to replace a faulty/underperforming service that implements automated activity in a BP. In this case the selection is restricted to certain business measures, and hence QoS criteria, which have to be optimized independently from other BP activities. Moreover, our selection approach is performed at more
granular level, group-based selection, which is customized to different customer classes of different service levels and therefore is more efficient and cost-effective.

VI. CONCLUSION AND FUTURE WORK

This paper has addressed the problem of Web service selection to dynamically replace a failed or underperforming Web service that implements automated activities in a WS-BPs. Particularly, it focuses on customizing the service selection to different groups of BP instances associated with certain characteristics including business and QoS properties. The proposed group-based Web service selection approach is based on the emerging trend where a Web service is offered with differentiated QoS properties and pricing schemes. In contrast to traditional selection techniques, our approach combines multiple service instance offerings from different providers to meet quality requirements of groups of BP instances so the overall service cost and utility are optimized. Furthermore, the approach is based on the use of constraint programming, modelling the service selection as a multi-criteria optimization problem based on different classes of BP instances grouped based on customer profile/class and service levels.

The evaluation used a constraint satisfaction solver for conducting experiments which compared our group-based service selection with an “one-size-fits-all” solution, where the selection tries to find one service instance that satisfies the criteria of all running process instances. Experimental results demonstrated that our group-based service selection approach shows better performance in terms of minimizing costs and maximizing QoS properties utility when compared to the traditional “one-size-fits-all” selection approach. Specifically, in terms of financial cost, our approach shows an average reduction of 65%; and in terms of the achieved utility, our approach achieves an average improvement of 30%.

Future work will consider QoS properties with different data types such as set and string data type. We also intend to conduct further experiments to compare our approach with the “stick to the provider” technique where all selected service offering instances must be from the same provider.

ACKNOWLEDGEMENT

NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.

REFERENCES