VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wānanga o te Ūpoko o te Ika a Māui

School of Engineering and Computer Science
Te Kura Mātai Pūkaha, Pūrorohiko

PO Box 600
Wellington
New Zealand

Tel: +64 4 463 5341
Fax: +64 4 463 5045
Internet: office@ecs.vuw.ac.nz

Multifaceted Web Service Composition

Alexandre Sawczuk da Silva

Supervisors: Dr. Hui Ma, Prof. Mengjie Zhang

May 11, 2015

Submitted in partial fulfilment of the requirements for PhD.

Abstract

A short description of the project goes here.

Contents

Figures

Chapter 1

Introduction

1.1 Problem Statement

As applications increasingly interact with the Web, the concept of Service-Oriented Architecture (SOA) [16] emerges as a popular solution. The key components of SOA are often Web services, which are functionality modules that provide operations accessible over the network via a standard communication protocol [6]. One of the greatest strengths of Web services is their modularity, because it allows the reuse of independent services that provide a desired operation as opposed to having to re-implement that functionality. The combination of multiple modular Web services to achieve a single, more complex task is known as *Web service composition*. At a basic level, services are combined according to the functionality they provide, i.e. the inputs required by their operations and the outputs produced after execution. Several services may be connected to each other, with the outputs of a service satisfying the inputs of the next, starting from the composition's given overall input and finally leading to the composition's required output.

An example of an application of Web service composition is ENVISION, an EU-funded project whose aim is to provide a framework for discovering and composing Web services that perform geospatial analysis on data, thus enabling environmental information to be more easily processed for research and decision-making purposes [13]. ENVISION is meant to be used by scientists who are experienced with geographic models but do not have a technical computing background, therefore the motivation of this work was to create a solution that is as simple to use as possible. The system offers a way to search for (discover) services that provide environmental data as well as processing services by using a word-based and coordinate-based search. Users create compositions by manually selecting and assembling a Business Process Model (BPM), which is later transformed into an engine-runnable representation. The environmental data and processing applications are packaged as services, meaning that they are easily available for reuse and thus contribute to the faster identification of important environmental trends.

Despite the usefulness of Web service composition, conducting such a process manually is fraught with difficulties. In order to illustrate these problems, an experiment was performed in which students with a good knowledge of programming and of Web services were asked to manually create Web service compositions to address a range of well defined real-world problems [12]. Results show that they faced a number of difficulties at different composition stages, from the discovery of potential services to their combination. During the discovery phase, the most popular search tools used by students were Web service portals and generic search engines. Authors highlighted that not a single discovery tool from the literature was used, because no single solution offers a large variety of services. This suggests that larger standardised service portals should be created, provided that the quality of the services offered is maintained. During the combination phase, students faced discrepancies between the concepts used by the interfaces of services: the terms used in the interfaces of any two services are often different from each other, even though those services may handle the exact same domain. Another problem is that the services used in a composition may produce too much data, or incur too much latency, meaning that the final composition is slowed down as a result. Standardising and automating the service composition process would eliminate the need for dealing with these difficulties manually, thus developing a system capable of creating compositions in a fully automated manner is one of the holy grails of the field [14].

Automated Web service composition is complex, and in fact it is considered an *NP-hard problem*, meaning that the solution to large tasks is not likely to be found with reasonable computation times [15]. Due to this complexity, a variety of strategies have been investigated in the literature, focusing on two fundamental composition approaches: *workflow-based approaches*, where the central idea is that the user provides an abstract business process which is to be completed using concrete services, and *AI planning-based approaches*, where this abstract process is not typically required and instead the focus is on discovering the connections between services that lead from the task's provided output to its desired [15]. In workflow-based approaches, the abstract functionality steps necessary to accomplish the task requested by the user have already been provided, so the objective is to select the concrete services with the best possible quality to fulfil each workflow step. The advantage of such approaches is that the selection of services may easily be translated into an optimisation problem where the objective is to achieve the best overall composition quality. This optimisation is often performed using *Evolutionary Computation* techniques. However, the disadvantage is that a workflow must have been already defined, which likely means that it has to be manually designed. Planning-based approaches, on the other hand, have the advantage of both determining the workflow to be used in the composition and selecting services to be used at each step, abiding by user constraints. The disadvantage of such approaches is that they are unable to also perform quality-based optimisation on the selected services.

The overall goal of this thesis is to propose a Web service composition approach that hybridises elements of AI planning-based approaches and Evolutionary Computation workflow-based approaches, enabling the construction of a workflow according to user constraints as well as the optimisation of that workflow according to the quality of its composing services. This new approach tackles several aspects of Web service composition, such as the use of multi-objective optimisation, semantic Web service selection, and dynamic Web service composition.

1.2 Motivations

The intricacy of Web service composition lies in the number of distinct facets it must simultaneously account for. As the **first facet**, services must be combined so that their operation inputs and outputs are properly linked, i.e. the output produced by a given service is usable as input by the next services in the composition, eventually leading up to the desired overall output. As the **second facet**, the composition must meet any specified user constraint. A constraint is defined as a user restriction that must be met in order for a composition solution to be considered valid, and this mainly concerns execution flow features (e.g. the composition must have multiple execution options – branches – according to a given condition) [20, 18, 8]. As the **third facet**, the composition must achieve the best possible overall Quality of Service (QoS) with regards to attributes such as the time required to execute the composite services, the financial cost of utilising the service modules, and the reliability of those modules. As the **fourth facet**, the environment of the composition must be recognised as dynamic, with QoS values that fluctuate and services that become unavailable/available over time. These facets are discussed in more detail below:

1. **Functionality:** Functional solutions can be obtained by using two main approaches: *immediate* and *gradual*. Whenever a solution is built using immediate approaches, this solution is fully functional, i.e. the inputs of all atomic services are fully satisfied, and so are the overall task outputs given the provided inputs. This is typically done by employing AI planning algorithms that verify the correctness and completeness of the connections between services at every building step [20]. When a solution is built using gradual approaches, on the other hand, there are no guarantees that it will be fully functional. As opposed to performing checks at every building step, gradual approaches rely on the notion of improving solutions over multiple iterations, each time penalising incorrect and/or incomplete connections between services. In practice, these approaches are implemented using Evolutionary Computation techniques with penalties enforced through the fitness function [17]. The simplest way of establishing that the inputs of an atomic service have been satisfied is by verifying that there is an exact type match between each target input and its corresponding incoming connection (i.e. the output of a previous service). However, a more sophisticated matching approach relies on measuring the semantic similarity between the output and input types in question with the help of an ontology [2]; the smaller the distance between the two values, the better the match.

2. **Conditional constraints:** User constraints and preferences are a common requirement when performing Web service composition. For example, consider the case of a user who wishes to book transportation and accommodation for a trip, including transportation from the airport to the hotel, a room at the hotel for a given number of days, and transportation back to the airport [3]. In this scenario, the user is likely to have certain conditional constraints; for example, if the hotel booking is for a two-star hotel, then the transportation to and from the aiport should be arranged using a taxi service, otherwise a shuttle service should be used. Different techniques have been explored to achieve compositions that consider such constraints, including the use of AI planning with rules encoding user constraints [2] and the representation of the composition as a constraint satisfaction problem to be processed with a solver engine [8]. This territory has not been widely explored by applying Evolutionary Computation (EC) techniques; yet, due to their flexibility and efficiency, it would be interesting to focus on the investigation of ways in which to extend them to apply these constraints. It must be noted that constraints relating to Quality of Service (QoS) values are not included in this facet.

3. **Quality of Service:** Quality of Service (QoS) refers to the non-functional (quality) attributes associated with a Web service, such as its expected execution time when answering requests, its availability, and its scalability [9]. The overall quality of compositions can be measured by aggregating the individual QoS values of its constituting atomic services, meaning that it is possible to optimise the composition's QoS by selecting the right combination of constituting services. As mentioned earlier, the selection of a set of services in order to maximise the overall QoS can be mapped into a classic optimisation scenario, and this has been done extensively in the literature. A variety of EC techniques have been used in this context [19], as well as other optimisation techniques such as integer linear programming [21]. Interestingly, AI planning

techniques were also used to this end, even though in this case they are incapable of also considering the conditional constraints facet [5]. In the realm of QoS optimisation, multi-objective techniques are often employed to the problem of Web service composition, since the optimisation of potentially conflicting QoS attributes such as time and cost is more intuitively performed using independent objective functions [11].

4. **Dynamic composition:** In a more realistic scenario, Web service compositions exist within a dynamic environment where the quality and availability of the atomic services in the repository varies as time passes. In such a dynamic environment, providing a static composition solution to a task is no longer enough, since this solution may decrease in quality throughout time and/or become non-executable if some of its composing services go offline. Thus, the focus of dynamic Web service composition is on monitoring and updating composition solutions as they become outdated [10]. The majority of techniques aimed at updating outdated or faulty compositions rely on building solutions that present constructs allowing for dynamic adaptation [1], but not many EC-based approaches have been tried in this area.

The above discussion references several techniques that have been proposed to address the composition problem. These techniques produce promising results, however they do not account foll all of these composition facets at once. For example, AI planning techniques for composition focus on guaranteeing functional correctness (first facet), and either fulfilling constraints (second facet) or optimising QoS (third facet); similarly, EC techniques such as Genetic Algorithms (GA) and Genetic Programming (GP) focus on QoS in addition to functional correctness, but do not include the fulfilment of constraints.

1.3 Research Goals

The overall goal of this thesis is to propose a hybrid Web service composition approach that considers elements from all the four facets described above when generating solutions. More specifically, this approach combines elements of AI planning, to ensure functional correctness and constraint fulfilment, and of Evolutionary Computation, to evolve a population of near-optimised solutions from a QoS standpoint. The research aims to determine a flexible way in which planning and EC can be combined to allow the creation of solutions to solve composition problems that require multiple execution paths. The work conducted in this thesis intends to address the following research questions:

(i) *What would be the components of a technique to create composition representations that allow for different execution paths to be optimised simultaneously with regards to quality?*
Traditionally, AI planning and EC have been employed separately to solve the problem of Web service composition. On the one hand, AI planning techniques are outstanding at ensuring the creation of composition solutions that have appropriate connections between the outputs and inputs of the composing services, and also at ensuring the creation of branches according to the constraints provided. On the other hand, EC techniques are ideal for encountering solutions with a good overall quality amongst a very large array of possibilities. Given the strengths of each set of techniques, it is ideal to combine these two composition strategies into a single approach that offers both sets of capabilities, and thus considers more service composition facets simultaneously. In this hybrid approach, users would be able to specify the conditions of when branching should occur, the order in which these conditions should be observed, and the outputs each execution branch should produce; the technique then returns a suitable solution.

(ii) *What is the best way of independently optimising the Quality of Service (QoS) attributes of a composition with multiple execution branches?*
Existing optimisation approaches have focused on optimising the QoS of composite services that have only one execution path, by relying either on a single or on multiple objective functions. In the case of single-objective approaches, the different QoS attributes are combined through a weighted sum that produces a unified quality score used for ranking candidate solutions; multi-objective approaches, on the other hand, evaluate each QoS attribute using a separate score and divide candidate solutions into groups by comparing each of these dimensions simultaneously. When optimising services with multiple execution paths, the quality of different branches must be considered, either by aggregating the QoS values of each branch or by considering them independently.

(iii) *What is the best semantic Web service selection method for a hybrid planning-EC technique?*
When building a candidate composition solution, atomic services must be selected according to the compatibility of their inputs. The simplest form of selection is when the inputs of services are matched according to their exact type, though recently more sophisticated semantic approaches that also allow for inexact matches have also been investigated. In a typical selection scenario, a concrete service is chosen to fulfil the functionality specified by an already-defined abstract service, which restricts the complexity of the problem. However, the concept of an abstract service cannot be used when selecting services during an AI planning-based composition, which decreases the efficiency of this approach. Thus, a better way of performing semantic selection must be designed for use with AI planning-based composition approaches.

(iv) *What are the adaptations necessary to make an EC-based composition technique suitable to a dynamic environment?*
Evolutionary Computation (EC) approaches to Web service composition have been extensively investigated in static scenarios, where the quality and availability of services are assumed to remain constant, however the use of these techniques has only been superficially explored in dynamic scenarios. Despite this lack of research, EC-based approaches show promise in the area of dynamic Web service composition for two reasons: firstly, they allow a composite service system to *self-heal* by maintaining a population of solutions and using them as alternative compositions in the case of failure; secondly, they support *dynamic adaptation* by allowing solutions to be further evolved in order to account for changes in the QoS values of atomic services. Given these advantages, it is beneficial to investigate the use of EC approaches in a dynamic composition context.

The research goal described and outlined above can be achieved by completing the following set of objectives, which are intended to be used as research guides throughout this project:

1. *Determine the best direct representation for the planning-EC composition technique.*
The creation of a hybrid composition technique that combines elements from planning and evolutionary computation requires a decision on the representation each composition solution should adopt. Undoubtedly the simplest model would be that of a linear vector, with each element being a service that should be included into the composition, however this linear structure does not satisfactorily encode the relationships and links between different services. Another problem with a vector is that the EC techniques

which use such a representation do not allow for structures of varying lengths, a requirement when performing fully automated composition. A tree or graph representation would be better suited to the task at hand, since such structures are naturally capable of representing the links between the composing services and also multiple execution branches. Choosing between these two structures presents some trade-offs that must be carefully considered, such as the existence of EC techniques that support that structure and the computational cost they incur.

2. *Develop an indirect representation for the planning-EC composition technique.*
   The structures discussed in Objective 1 may be referred to as *direct representations* of solutions, since the genotype and the phenotype of each solution are the same, meaning that solutions are represented directly as they should be interpreted. Despite the intuitiveness of direct representations, it has been argued in the problem domain of scheduling that indirect representations, where the final result must be decoded from a population candidate, have been shown to have better search performance than their direct counterparts [7]. Due to this evidence, an indirect candidate representation should be proposed and compared to the direct representation. This indirect representation could be based on the Web service composition Particle Swarm Optimisation (PSO) approach proposed in [4].

3. *Develop a hybrid representation for the planning-EC composition technique.*
   In case there is a trade-off between the execution time and the quality of the solutions produced by the direct and indirect representations developed in Objectives 1 and 2, a hybrid representation should be developed to combine the strengths of these two approaches. This representation could comprise the structure used in the direct representation with the addition of weights from the indirect representation. Comparisons should be performed to determine whether the hybrid solution presents any performance or quality gains.

4. *Propose a many-objective approach to optimise the quality of candidates with multiple execution branches.*
   Two factors must be taken into account when optimising Web service compositions with multiple execution branches: firstly, each Quality of Service (QoS) attribute must be optimised independently, since they may be conflicting to each other; secondly, each branch of the composition must also be independently considered, since the QoS values for each branch may vary significantly despite their common root. Given these two factors, a many-objective technique should be employed in order to optimise the quality of solutions while at the same time considering these two factors independently.

5. *Develop an EC technique for performing semantic Web service selection in the context of a planning-based composition technique.*
   As discussed in Objective 1, the composition technique investigated in this thesis combines elements of AI planning and of EC approaches to achieve compositions that are correct, have conditional branching, and can be optimised according to their QoS attributes. In planning-based approaches to service composition, however, the semantic selection process of candidate atomic services is inefficient, since there are many possible service matches to consider at each planning step. In fact, it may be unfeasible to consider all possible service matches in an exhaustive manner when handling larger service repositories, which invites the use of non-exhaustive approaches such as EC techniques to accomplish this task. Thus, this objective entails the use of an optimisation to discover semantically matching services to be included in a composition candi-

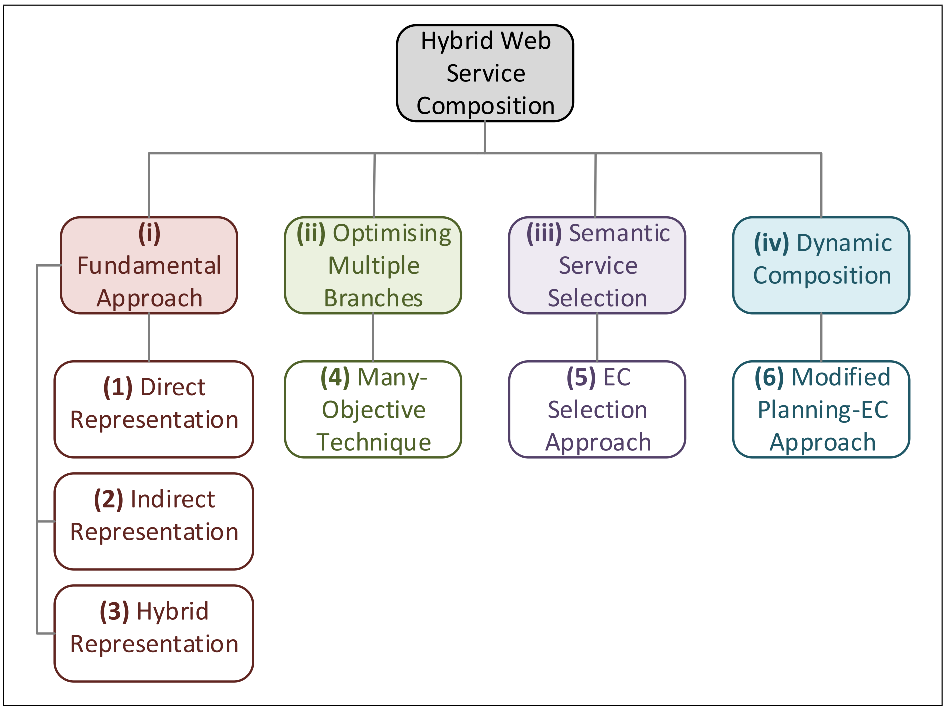
Hybrid Web Service Composition
(i) Fundamental Approach
(ii) Optimising Multiple Branches
(iii) Semantic Service Selection
(iv) Dynamic Composition
(1) Direct Representation
(2) Indirect Representation
(3) Hybrid Representation
(4) Many-Objective Technique
(5) EC Selection Approach
(6) Modified Planning-EC Approach

Figure 1.1: Research questions and their corresponding objectives.

date. Note that this objective aims to investigate the use of an optimisation technique nested within the service selection step of the planning-EC approach.

6. *Modify the planning-EC composition technique to work in a dynamic environment.*
   In a static scenario the planning-EC technique is executed once for a given composition task, returning a composite result under the assumption that the quality levels and the availability of the atomic services included in that result will remain constant. In a more realistic dynamic scenario, however, the quality of the services in the repository may fluctuate and occasionally services may become unavailable. To account for these setbacks, solutions must be corrected and updated in response to changes in the environment. In order to do this, the planning-EC technique is to be modified to retain a population of candidates as alternatives in case of failure, and further generations are to be evolved as the QoS values of services in the repository change, thus leveraging the natural features of Evolutionary Computation.

As shown in Figure 1.1, Objectives 1, 2, and 3 are the core of this project, and answer research question (i) by providing the basis for an optimisation technique that admits Web service composition solutions with multiple execution branches. Objective 4, on the other hand, answers research question (ii), which aims to determine the ways in which the branches of a solution with conditional constraints can be independently optimised. Objective 5 answers research question (iii), leading to the investigation of a semantic selection approach for planning-based composition techniques. Finally, Objective 6 answers question (iv) by proposing one way in which EC-based techniques can be used in a dynamic Web service composition scenario.

1.4 Published Papers

1.5 Organisation of Proposal

The remainder of the proposal is organised as follows: Chapter ...; Chapter ...; Chapter ....

Bibliography

[1] Alférez, G. H., Pelechano, V., Mazo, R., Salinesi, C., and Diaz, D. Dynamic adaptation of service compositions with variability models. *Journal of Systems and Software 91* (2014), 24–47.

[2] Boustil, A., Maamri, R., and Sahnoun, Z. A semantic selection approach for composite web services using OWL-DL and rules. *Service Oriented Computing and Applications 8*, 3 (2014), 221–238.

[3] Boustil, A., Sabouret, N., and Maamri, R. Web services composition handling user constraints: towards a semantic approach. In *Proceedings of the 12th International Conference on Information Integration and Web-based Applications & Services* (2010), ACM, pp. 913–916.

[4] da Silva, A. S., Ma, H., and Zhang, M. A graph-based particle swarm optimisation approach to qos-aware web service composition and selection. In *Evolutionary Computation (CEC), 2014 IEEE Congress on* (2014), IEEE, pp. 3127–3134.

[5] Deng, S., Wu, B., Yin, J., and Wu, Z. Efficient planning for top-k web service composition. *Knowledge and information systems 36*, 3 (2013), 579–605.

[6] Gottschalk, K., Graham, S., Kreger, H., and Snell, J. Introduction to web services architecture. *IBM systems Journal 41*, 2 (2002), 170–177.

[7] Hart, E., Ross, P., and Corne, D. Evolutionary scheduling: A review. *Genetic Programming and Evolvable Machines 6*, 2 (2005), 191–220.

[8] Karakoc, E., and Senkul, P. Composing semantic web services under constraints. *Expert Systems with Applications 36*, 8 (2009), 11021–11029.

[9] Ko, J. M., Kim, C. O., and Kwon, I.-H. Quality-of-service oriented web service composition algorithm and planning architecture. *Journal of Systems and Software 81*, 11 (2008), 2079–2090.

[10] Li, G., Liao, L., Song, D., and Zheng, Z. A fault-tolerant framework for qos-aware web service composition via case-based reasoning. *International Journal of Web and Grid Services 10*, 1 (2014), 80–99.

[11] Liu, S., Liu, Y., Jing, N., Tang, G., and Tang, Y. A dynamic web service selection strategy with qos global optimization based on multi-objective genetic algorithm. In *Grid and Cooperative Computing-GCC 2005*. Springer, 2005, pp. 84–89.

[12] Lu, J., Yu, Y., Roy, D., and Saha, D. Web service composition: A reality check. In *Web Information Systems Engineering–WISE 2007*. Springer, 2007, pp. 523–532.

[13] MAUÉ, P., AND ROMAN, D. The envision environmental portal and services infrastructure. In *Environmental Software Systems. Frameworks of eEnvironment*. Springer, 2011, pp. 280–294.

[14] MILANOVIC, N., AND MALEK, M. Current solutions for web service composition. *IEEE Internet Computing 8*, 6 (2004), 51–59.

[15] MOGHADDAM, M., AND DAVIS, J. G. Service selection in web service composition: A comparative review of existing approaches. In *Web Services Foundations*. Springer, 2014, pp. 321–346.

[16] PERREY, R., AND LYCETT, M. Service-oriented architecture. In *Applications and the Internet Workshops, 2003. Proceedings. 2003 Symposium on* (2003), IEEE, pp. 116–119.

[17] RODRIGUEZ-MIER, P., MUCIENTES, M., LAMA, M., AND COUTO, M. I. Composition of web services through genetic programming. *Evolutionary Intelligence 3*, 3-4 (2010), 171–186.

[18] SOHRABI, S., PROKOSHYNA, N., AND MCILRAITH, S. A. Web service composition via the customization of golog programs with user preferences. In *Conceptual Modeling: Foundations and Applications*. Springer, 2009, pp. 319–334.

[19] WANG, L., SHEN, J., AND YONG, J. A survey on bio-inspired algorithms for web service composition. In *Computer Supported Cooperative Work in Design (CSCWD), 2012 IEEE 16th International Conference on* (2012), IEEE, pp. 569–574.

[20] WANG, P., DING, Z., JIANG, C., AND ZHOU, M. Automated web service composition supporting conditional branch structures. *Enterprise Information Systems 8*, 1 (2014), 121–146.

[21] YOO, J. J.-W., KUMARA, S., LEE, D., AND OH, S.-C. A web service composition framework using integer programming with non-functional objectives and constraints. *algorithms 1* (2008), 7.