University of Alberta

A Multi-Attribute Service Composition Model in a Dynamic Environment

by

Abhishek Srivastava

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Department of Computing Science

Fall 2011
Edmonton, Alberta

To my parents and sister

Abstract

Dynamic service composition involves the run-time selection of service elements that are combined to form a larger more complex composite application. There are several issues in dynamic service composition that are examined in this thesis. First, the existence of a large number of service elements with similar or identical functionalities makes it difficult to select appropriate service elements dynamically. To address this issue, we utilize the non-functional attributes of the service elements. The non-functional attribute values are discriminating factors on which one service element is deemed better than other candidates for the specific composition. We first present novel selection techniques based on single attribute criteria for the three non-functional attributes: *reliability, waiting-time, and reputation.*

More generally, however, there are usually a number of service attributes, which include non-functional and functional attributes, associated with each service element. In addition, the customers who are the ultimate stakeholders of the composition process have varied preferences for these service attributes. To incorporate simultaneously all the service attributes and the preference weights of service customers, we introduce a unifying factor called *affinity* that is a function of all the service attributes of the service elements and the preferences articulated by customers. The affinity factor is embedded in a new model called the Affinity Model that is used for the dynamic selection of service elements to form service compositions. The Affinity Model utilizes the affinity values calculated to select service elements following a Greedy algorithm where the service elements with the largest affinity values are selected for each functionality.

The efficacy of the Affinity Model is validated by simulating the service composition process as a game called the *Ambitious-Traveler*. The validation procedure

involves a set of human participants who volunteer to play the Ambitious-Traveler game. The game is simultaneously played in an automated manner using selection decisions made by the Affinity Model. The results show a comparable or superior performance by the Affinity Model to that of the human players in 90% of the trials. This validates the hypothesis that the Affinity Model is capable of making service selections that are comparable to or better than the intuitive judgement of humans.

Acknowledgements

I am deeply indebted to my supervisor Dr Paul Sorenson for all his guidance and support during the course of this thesis. Dr Sorenson has been a source of constant motivation through these five years especially when the chips were down. His unique style of supervision would bring out the best in me without making me feel overly stressed and his tremendous insight on the subject always ensured that my research progressed in a fruitful direction. I will always cherish the time spent working with him.

I would like to thank Dr Eleni Stroulia who as part of my supervisory committee gave me very useful feedback. I am presently collaborating with her on research subjects related to my thesis and am thoroughly enjoying it. I would like to thank Dr Ken Wong who was also part of my supervisory committee and gave useful suggestions especially during the weekly Software Engineering Research Laboratory meetings and during my candidacy and final oral examination. Thanks are due to Dr Hausi Müller, Dr Raymond Patterson, and Dr Ehab Elmallah who were part of my examining committee at different stages of my thesis. Their feedback was very useful and has significantly contributed towards the improvement of my thesis. I would also like to thank Dr James Hoover, Dr Osmar Zaïane, and Dr Guohui Lin for useful advice and suggestions during the course of my Ph.D.

I would like to acknowledge the support of Zhijie Wang who started off being by office mate and over the years evolved into a very close friend. We started working on our Ph.Ds together and he has been a constant support through the years especially when circumstances were tough. I would like to thank Xian Chen, my colleague and friend. As a participant of our weekly research meetings, his suggestions were useful and positively influenced the course of my thesis. I would also

like to thank Sunil Ravinder, Sajib Barua, Amit Satsangi, Himanshu Vashishtha, Ken Bauer, Sudip Barman, and Qing Dou. I will always cherish their friendship and support. Also, a very big thank you to my brother-in-law, Mr Ravi Sahay, who has been extremely supportive during the course of my Ph.D.

Big thanks to my lovely wife Smriti who was by my side during the conclusive and possibly toughest part of this thesis. She would bear the long working hours and sometimes distressing circumstances always with a smile on her face.

Incidentally on the day I submitted this thesis to the examination committee, we were blessed with our bundle of joy, our son Archit. His arrival has been a lucky charm for us and simply looking at him eases away tensions and makes life easier.

Finally, I dedicate this thesis to my parents and sister. Without their love and support, this thesis would not have been possible. They have seen me through smiles and tears and have been pillars of strength for me. My father, himself a researcher, has been a source of constant motivation. I draw huge inspiration from his accomplishments that have come despite all odds.

Above all, I thank the Almighty whose blessings have made all this possible.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Service composition is the combination of a set of service elements to form a larger composite service that can be used to offer more complex functionality. This composite service can be further combined in a recursive manner with other services to form even larger services. The Software Services and Systems Network (S-Cube) [1], a project with the aim of establishing a multidisciplinary European research community, defines service composition as:

"Service Composition is a combination of a set of services for achieving a certain purpose. Different service composition types can be distinguished, in particular: service orchestration, service choreography, service wiring, and service coordination."

We choose to include this definition here not only because it is comprehensive but also because the S-Cube is involved in cutting edge research on services and their composition. A good overview of service composition can be found in the much cited work of Srivastava *et al.* [2] The benefits of service composition are: 1) it provides quick solutions to complex requirements, 2) it promotes reuse of existing capabilities, and 3) it yields solutions that are easily modifiable and flexible. The driving forces behind forming service compositions are traditionally the availability (or the lack) of resources, the functionality that needs to be achieved by the composition, and the business interests (e.g. return on investment) of service providers putting together the composition. Two factors that are important while forming service compositions are: 1) non-functional (quality of service) attributes of the service composition, 2) the preferences of the customers for whom the composition is formed.

One of the major contributions of this thesis is that it examines how the two factors mentioned can be effectively incorporated into the service composition process. There is significant existing literature that puts forth the idea of using non-functional attributes for forming compositions. The thesis addresses some of the important limitations in the existing methods and reinforces the use of non-functional attributes in service composition by proposing novel techniques for the same.

Perhaps more importantly, the second idea of using the customer preferences in forming effective compositions is relatively unexplored. We propose a technique

that incorporates the direct involvement of the customer in the service composition process.

The remainder of this chapter is organized as follows: Section 1.1 provides an overview of the service composition process and classifies service composition on the basis of the procedure followed in forming the composition. The section also identifies the objectives of the thesis and the area in service composition that we address. Section 1.2 outlines the structure of the remainder of the thesis.

1.1 Service composition and its types

Service composition as defined earlier is the combination of a set of service elements to form a larger more complex service. In general, service composition comprises of a set of abstract capabilities or functionalities that are instantiated by real service elements. For every abstract functionality there are usually a number of real service element instances available. The first step of service composition comprises selecting the appropriate instance for each abstract functionality. This is shown in Figure 1.1. The circles represent the abstract functionality and the rectangles represent the real service elements that are capable of instantiating the functionalities.

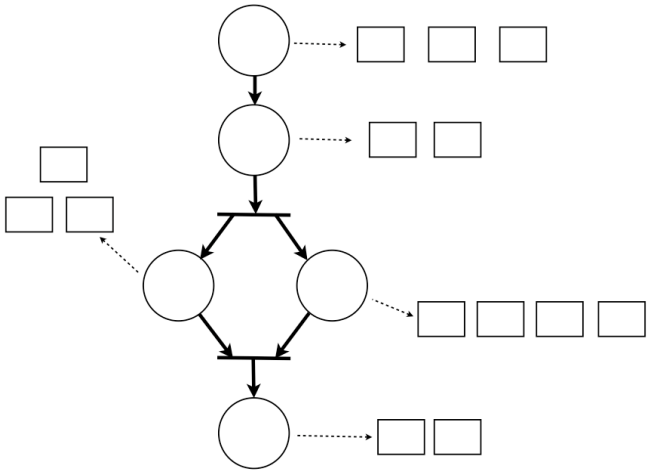


Figure 1.1: Dynamic service composition: abstract services and the instances

After the service elements instantiating the abstract functionalities are selected, the next step in service composition is 'chaining' the selected service elements together. This mainly involves orchestrating the messages passed between the selected service elements. The *Business Process Execution Language (BPEL)* [3] is typically used for this purpose. Alternatively, the 'request-response' messages between service pairs are regulated through a process called 'choreography' and this forms the basis of chaining the selected service elements together. The *Web Service Conversation Language (WSCL)* [4] is one of the technologies used for

choreographing web services, although it is not nearly as established as an industry standard as BPEL.

In the thesis we concentrate on the first step of service composition: the selection of service elements to instantiate the abstract functionalities. We classify service composition into different types based on the procedure followed to select the service elements. The service composition types are distinguished as: 1) static and dynamic composition, 2) composition with service selections based on functional and non-functional attributes, 3) composition with service selections based on single and multi attributes, and 4) composition with and without direct customer involvement. These composition types are described in the following subsections.

1.1.1 Static and dynamic composition

As the name implies static composition is one in which the selection of service elements that instantiate the abstract functionalities is done statically in advance of the deployment of a service. Only after all the service elements taking part in the composition are determined is the step of chaining together the service elements to form the composition initiated. Service orchestration and choreography discussed earlier are procedures followed for static composition.

Dynamic composition on the other hand leaves the task of service selection to run-time. The service elements are selected only at the point at which the capability of the service element is required while the process of chaining together service elements is already under way. Figure 1.2 shows the first dimension of our classification of the types of service compositions: static and dynamic. A comprehensive discussion on static and dynamic compositions can be found in Bucchiarone *et al.* [5]. In this thesis, we deal with dynamic service compositions which brings us to the **first objective** of the thesis: *to devise a model that facilitates the composition of service elements dynamically.*

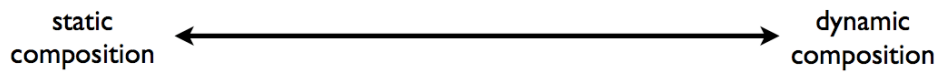


Figure 1.2: Classification of service composition: static and dynamic compositions

1.1.2 Composition with service selections based on functional and non-functional attributes

Service composition with selections based on functional attributes is the approach traditionally emphasized in composition strategies. The service element that is selected for an abstract functionality is the one that can comprehensively cater to the functionality requirements of the customer.

An exponential rise in the number and kinds of service elements available in recent years has led to the nice problem of too much choice. For every functionality we seek, there is a large pool of service elements each capable of performing the required functionality well. In such a scenario, service composition is done based on the non-functional attributes. The best among functional equals is the service element that conforms most closely to the non-functional requirements of the customer. Shuping Ran undertook pioneering work on the use of non-functional attributes in service selection [33]. Figure 1.3 shows compositions based on functional and non-functional attributes as the second dimension of service composition types in addition to static and dynamic compositions. In this thesis, we examine service composition mainly based on non-functional attributes.

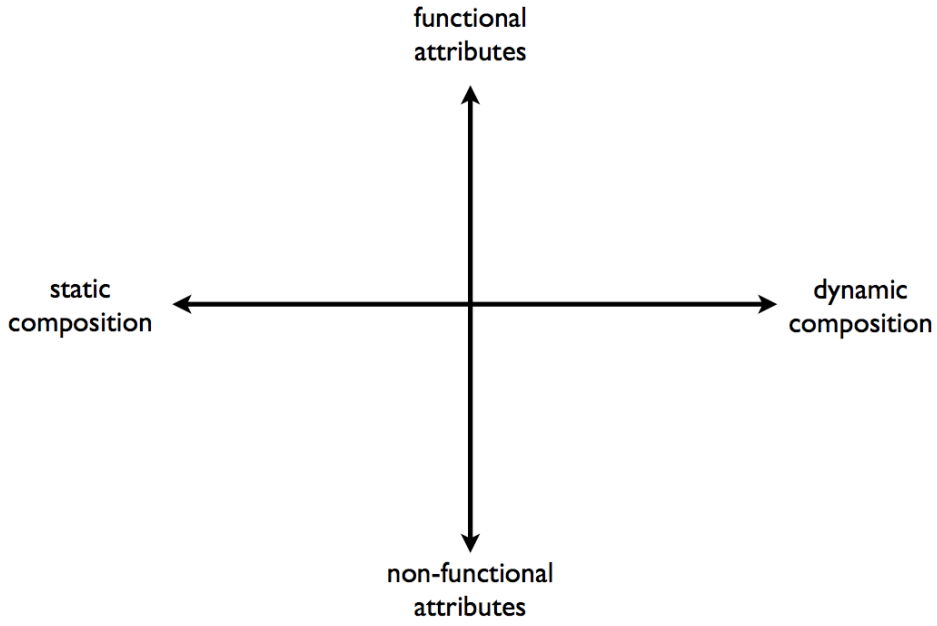


Figure 1.3: Composition based on functional and non-functional attributes

The **second objective** of the thesis is: *to devise and validate novel techniques for dynamic service composition based on non-functional attributes*. The non-functional attributes that we focus on in this thesis are: reliability, waiting-time, and reputation.

We choose these three non-functional attributes because they are important and unrelated attributes and can be looked upon as representative of most non-functional attributes in general. More specifically: reliability is arguably one of the most important non-functional attributes from the customer point of view; waiting-time is slated to gain in prominence with the rapid proliferation of services and customers vying for these services; and reputation gives us another angle of assessment of customer preferences as we assess service reputation on the basis of customer feedback.

1.1.3 Composition with service selections based on single and multi attributes

Composition based on non-functional attributes is further divided into two categories: compositions with service selections based on single attribute and multi attributes. In single attribute composition there is a single non-functional attribute based on which services are selected and composed. Multi-attribute service composition makes this problem more complex because now the selection of service elements for the composition is done on the basis of more than one non-functional attribute. The service elements that are selected need to have a balance of several non-functional attributes based on customer preferences. This categorization is shown in Figure 1.4.

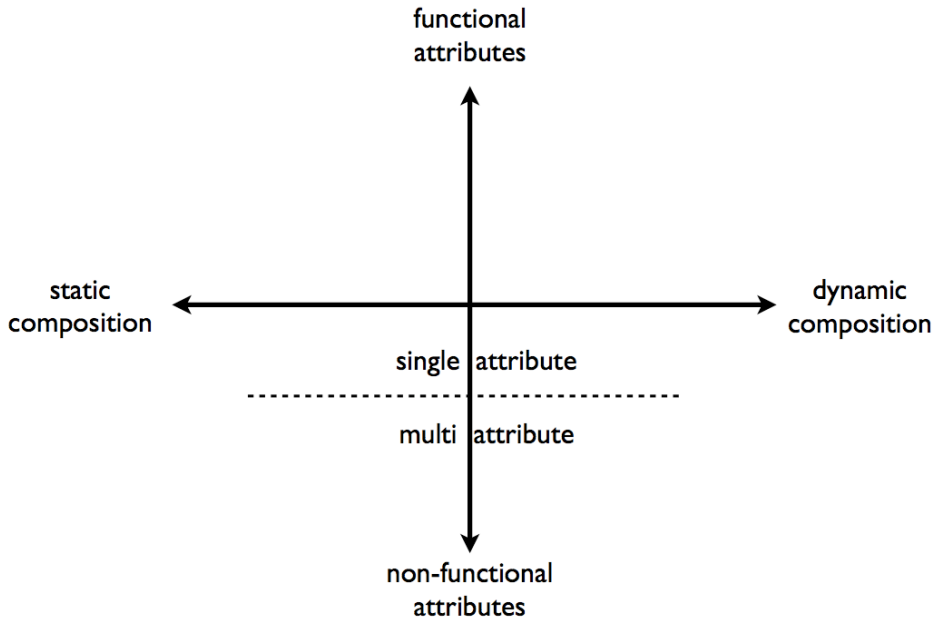


Figure 1.4: Composition based on single attribute and multi attributes

This leads to the **third objective** of the thesis: *to incorporate multiple non-functional attributes simultaneously into the process of dynamic service composition.*

1.1.4 Composition with and without customer involvement

The next dimension of service composition is determined on the basis of whether there is active involvement of the customer in the service composition process or not. In theory every composition is formed in one way or another to meet the requirements of the customer. However, few composition processes allow active involvement of customers where the latter is directly influencing the selection of service elements at the various abstract functionality levels. Figure 1.5 shows the third dimension in service composition types along with the other two.

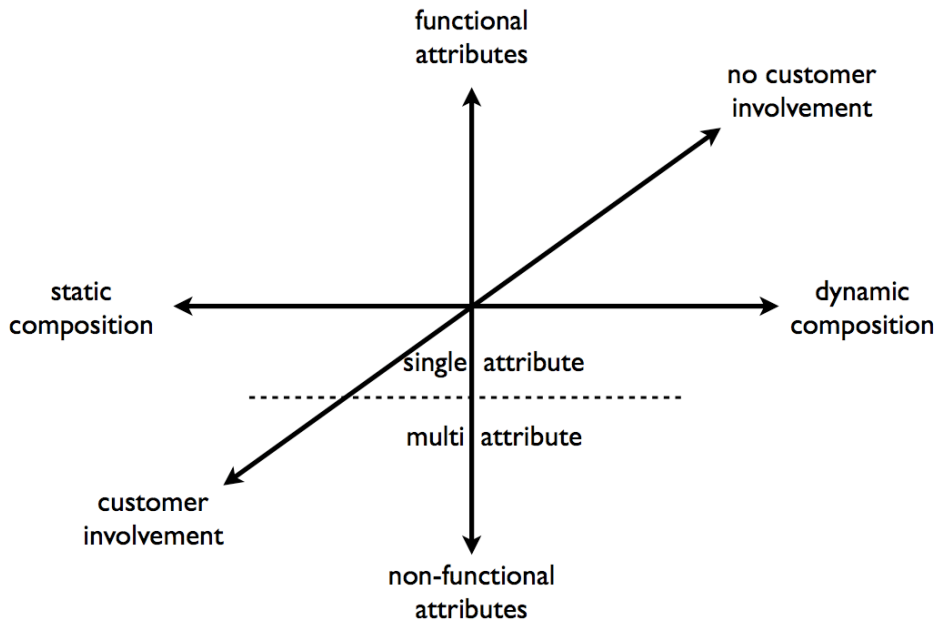


Figure 1.5: Classification of service composition: presence and absence of customer involvement

The **final objective** of the thesis is: *to devise a procedure that facilitates direct involvement of customer in the process of dynamic service composition such that the service composition formed closely conforms to the preferences of the customer.*

1.1.5 The focus of the thesis

In our work, we concentrate on the block shown in Figure 1.6. We propose a service composition model that does service selections *dynamically* based on *multiple non-functional attributes* of the service elements with the *direct involvement of customers* in the selection process.

Our approach is first to propose service composition techniques that are dynamic in nature and are based on the values of non-functional attributes of the service elements. We do not at this stage incorporate customer involvement as a major focus in the service composition process. The non-functional attributes based on which the service elements are selected are: reliability, waiting-time, and reputation. Each of these attributes is initially treated independently in the proposed techniques. We also include the cost attribute in our discussions but do not examine this attribute in detail in the thesis.

Subsequently, customer involvement in the service composition process is introduced. The customer involvement is expressed as the preferences that the customers have for the non-functional attributes in addition to the basic functionality of the service. All these factors, the non-functional service attributes, and the customer preferences are combined at run-time to give a dynamic multi-attribute service composition process with direct customer involvement thus fulfilling the four objectives

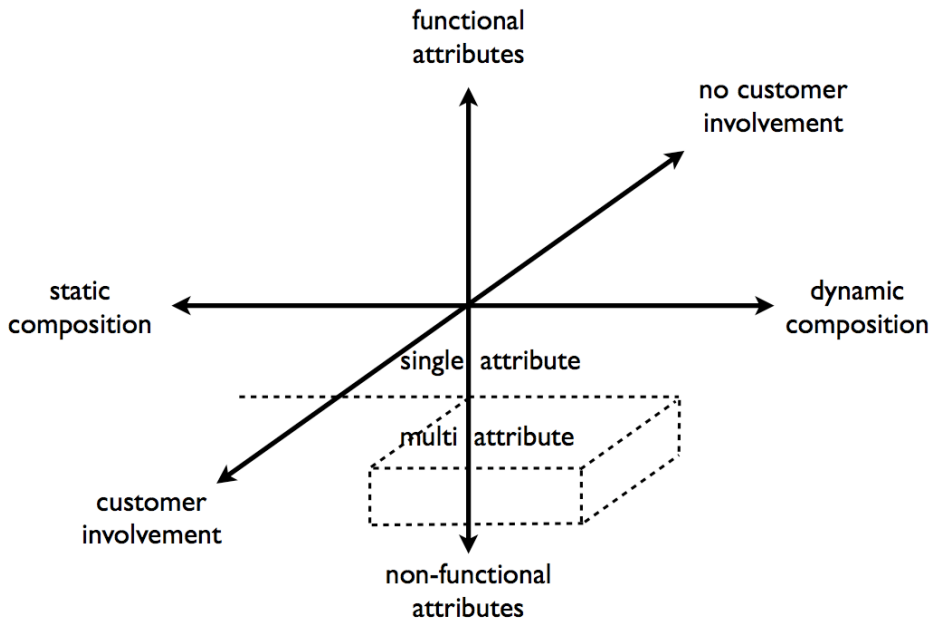


Figure 1.6: Focus of the thesis

outlined in the previous section. The process in shown in Figure 1.7.

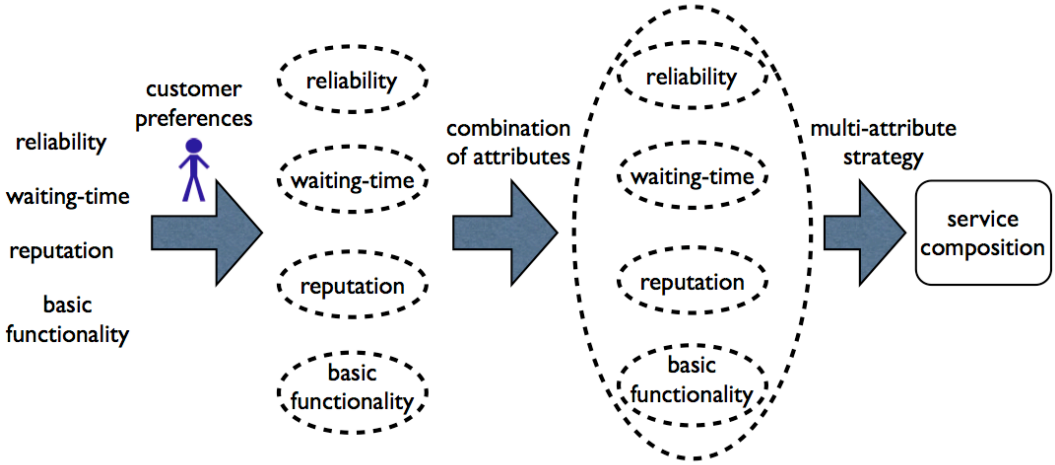


Figure 1.7: The proposed service composition approach

1.2 Structure of the thesis

The remainder of the thesis is structured as follows: Chapter 2 comprises a comprehensive discussion of literature related to our work. In addition, background information that sets the stage for the remainder of the thesis is included.

The next three chapters include novel techniques for service composition that are dynamic and based on the non-functional attributes. The customer involvement dimension is not emphasized in these chapters. Chapter 3 is a discussion on a technique we developed for dynamic service composition using the non-functional attribute reliability. The waiting-time attribute for putting together service compositions is examined in Chapter 4. Finally, Chapter 5 comprises a novel technique to utilize the reputation attribute for service composition.

Chapter 6 brings the customer involvement dimension into the composition process and also discusses how the three non-functional attributes of the earlier chapters are combined to create a multi-attribute service composition process.

Chapter 7 describes our validation procedure for the proposed service composition model. The validation approach comprises of a simple game called the *Ambitious Traveler* which is simultaneously played by human volunteers and automatically based on service selections made by the service composition model. A performance by the composition model that is comparable to that of the human volunteers serves to validate its efficacy.

Finally Chapter 8 summarizes the thesis contributions and describes some directions for future research.

Chapter 2

Background and related work

In this chapter, in Section 2.1, we discuss the concepts of services and service attributes. The latter is an important concept in this thesis and an introduction at this stage helps in understanding the techniques and procedures presented later.

In Section 2.2 we discuss related work on dynamic service compositions. There is a significant body of literature on dynamic service compositions wherein varied types of composition technique have been proposed for run-time service selection and composition. Many of these composition techniques are effective and have become accepted industry standards. During our explanation of existing techniques we relate these findings with the service composition approach presented in the thesis.

Finally in Section 2.3, we provide an overview of our service composition approach. The various steps in the composition procedure are discussed briefly to give the reader a high-level understanding of the working of the model. This sets the stage for the subsequent chapters where the steps introduced here are dealt with in detail.

2.1 Services and service attributes

A Service is defined by the World Wide Web Consortium (W3C) [9], an international community devoted to developing web standards, as:

"A service is a set of actions that form a coherent whole from the point of view of service providers and service requesters. The requester derives value from this 'set of actions' and in turn compensates the provider monetarily or otherwise for providing this value."

Service is a broad term and encompasses a large number of categories. Examples of service areas include hotels, restaurants, airlines, and hospitals. Each of these sectors has a well-understood pricing model and provides value to customers. From an Information Technology (IT) perspective, the advent of the Internet and its rapid adoption in our daily lives has had an impact on the delivery mechanism of services. Services are packaged into a form called web-services [10] that can

be delivered to customers using the medium of the Internet. Both service providers and customers benefit from the convenience and cost-effectiveness offered by web services.

With such rapid proliferation, services form an important sector of the economy. Nations around the world are swiftly moving towards becoming service based economies [11]. In fact, over 60% of the economies of the so called 'industrialized' nations are service based. This does not mean however that services are *replacing* products. Services are in fact complimenting products in a way that has been aptly described as "servitization" of products [12], where services play a supporting role to products to improve the overall customer experience. The supporting role could be in the form of 'service centers', 'customer support', 'technical assistance', etc.

2.1.1 Service attributes

A service has an associated set of attributes that characterize the nature of the service. Attributes are characteristic features of the service that define a distinct identity to each service. They express the capabilities of a service and help determine the suitability of a service for carrying out a certain task. Service attributes, depending on their nature, may or may not be expressible quantitatively. Quantifiable attributes are expressed using exact values and help characterize the nature of the service precisely. Non-quantifiable attributes on the other hand are expressed in relative terms using adjectives such as "high" and "low" or "big" and "small" and 'roughly' convey the nature of the service. We identify two important categorizations of service attributes pertinent to the contents of this thesis: 1) functional and non-functional attributes, and 2) static and dynamic attributes.

Functional and non-functional attributes

Functional attributes of a service are specific to the service and express the basic capabilities of the service. They define the functions that the service is *supposed* to perform. A service that fails to perform successfully these functions is deemed inoperable. An example of a functional attribute is: the money dispensing capability of a banking service.

A non-functional attribute of a service is a generic attribute and applies to any service. Non-functional attributes are characteristics of the service that express its quality. A service that has good values for the non-functional attributes is deemed to be of good quality and vice-versa. An example of a non-functional attribute is reliability. Every service irrespective of its type can have a reliability figure associated with it. High reliability values of a service contribute to its quality.

Cost is an interesting attribute that sits somewhere in between being a functional and a non-functional attribute. Cost can be called non-functional because it can be associated with any and all services. Every service has a cost, direct or indirect. On the other hand, cost is often looked upon as an attribute that expresses some notion

of value for the functional capability of a service. Therefore it is often viewed as a functionally related service attribute.

Static and dynamic attributes

The other important categorization of service attributes is: static and dynamic attributes. Static service attributes are attributes whose values remain fixed over an extended period of time. Static attribute values are either independent or dependent upon factors that remain constant over time. An example of a static service attribute is the security attribute of a banking service. This attribute value typically remains constant unless there is an upgrade of the system.

A dynamic service attribute on the other hand is one whose value gets updated frequently. These attributes are generally not dependent on a fixed set of factors. Dynamic service attributes are more expensive to measure and require continuous monitoring. An example of a dynamic service attribute is the waiting time at a banking service.

2.2 Dynamic service composition

Dynamic service composition involves the run-time selection of service elements that partake in the formation of a composite service. There are several approaches to dynamic service composition. Based on the work of Alamari *et al.* [14] dynamic service composition approaches are classified into four major groups: 1) wrapper based composition, 2) runtime service adaptation, 3) semantic service composition, and 4) declarative composition.

2.2.1 Wrapper-based composition

Wrapper-based composition is an approach to service composition in which two or more service elements dynamically combine to form a service 'wrap'. Such service wraps are characterized by a common interface through which the wrapped services interact with other external services. The purpose of forming service wraps is to allow service elements to interact with other services in an otherwise foreign environment. Such wraps are usually formed dynamically when the need arises and are a quick-fix substitute for complex service re-engineering tasks. Wrapper-based composition can be recursive in nature with service wraps joining with other services and/or other service wraps to form larger wraps. Figure 2.1 depicts a wrap of four service elements. Service element A is the common interface through which services B, C, and D interact with other external services.

A good example of wrapper-based composition is the *Proteus* system [15]. The system is brought into use in a situation where service elements that are not capable of processing XML commands need to interact with web-services. XML based messages are the normal mode of communication among web-services. The Proteus

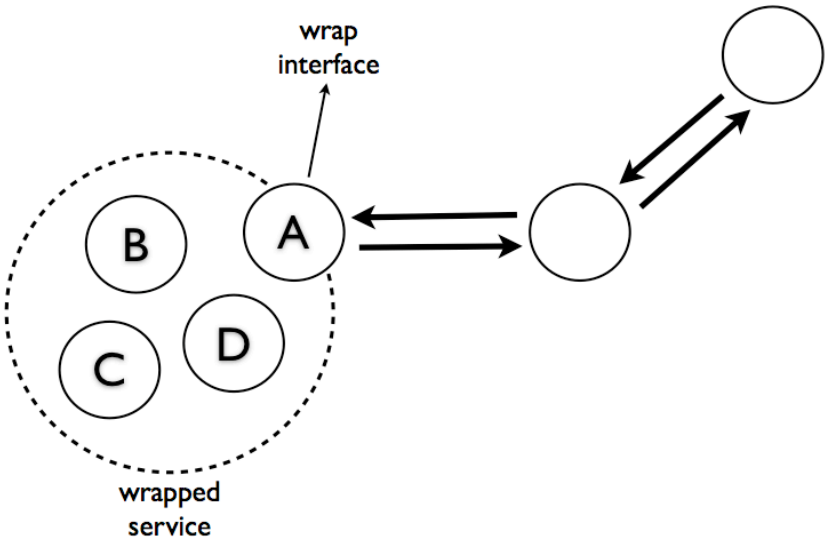


Figure 2.1: Wrapper based composition

system forms service wraps of these services and provides them with an interface that is capable of interactions in XML. The service wraps behave like web-services and can communicate effectively with other web-services.

2.2.2 Run-time service adaptation

Runtime service adaptation entails the dynamic addition or removal of capabilities to and from a service element. The main motivation for this is to enable the composition of otherwise incompatible services. To achieve this, several 'adaptation components' are developed in advance anticipating possible composition scenarios. These adaptation components are plugged into the service element at run-time when the need arises. Figure 2.2 shows a scenario where service elements A and B are unable to interact and form a composition due to possibly technical incompatibilities. An adaptation component is subsequently plugged into A to form a slightly modified service element A'. This service element with the additional capabilities is able to communicate with B

There are several limitations in this approach. First, substantial effort is typically needed to develop a large number of adaptation components many of which are never used. Second, the specific internal workings of the service element need to be known so that the adaptation components can be properly built and easily plugged in. Third, the efficiency of the service element is often compromised when several such adaptation plug-ins are incorporated. *Superimposition* is an example of runtime service adaptation [16]. Superimposition involves a reusable base component on which adaptation layers can be added as the need arises. There can be situations where multiple layers of functionality are added one on top of the other.

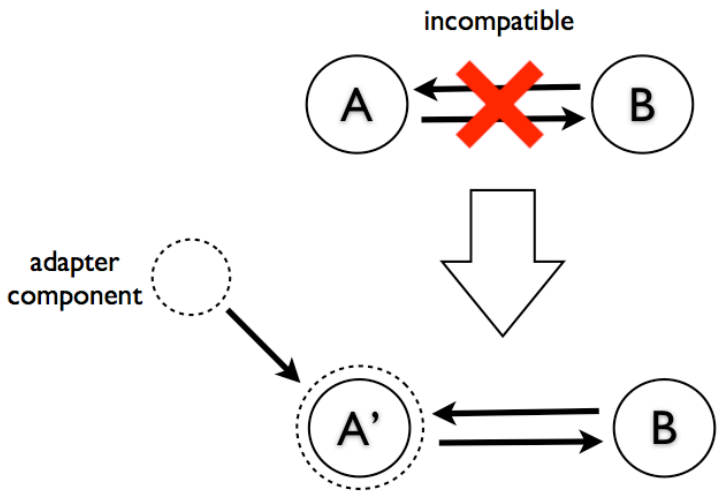


Figure 2.2: Run-time service adaptation

Superimposition is used for catering to required functionalities as well as to enable combination of incompatible components.

Wrapper-based service composition and runtime service adaptation are both interesting and widely discussed procedures for dynamic service composition. This is why they have been included in this section. They are however not easily applied to the service composition approach presented in this thesis. Wrapper-based and runtime adaptation techniques do not deal with the issue of selecting service elements for composition. They assume the appropriate service elements are already selected and comprise techniques of putting these together. Our approach on the other hand is involved mostly with the selection of appropriate service elements for composition. The remaining two service composition approaches discussed in this section are closer to our work and deal mainly with the selection of services for composition.

2.2.3 Semantic service composition

Semantic description of services enables automated and dynamic selection of services for composition. Semantic descriptions of services imply meta-data associated with the services that are usually expressed in XML and are hence machine readable. The meta-data includes information such as functionality of the services, input and output parameters of the services, preconditions, effects, etc. This information enables finding services that closely match the requirements of the composition. Furthermore, semantic descriptions of services also include service ontologies. The ontologies for different service domains enable the articulation of relations that exist between different kinds of services and make the matching of service elements with requirements even more efficient.

There is significant work on the use of semantic descriptions for dynamic web

service composition. Lecue *et al.* [17] examine the similarities between output parameters of service elements and the input parameters of potential service elements. A high degree of similarity between these indicates a good match. Semantic descriptions of the services are utilized to gather information on the output and input parameters of services. Arpinar *et al.* [18] also follow a similar procedure and rank potential service elements based on the degree of parameter similarity.

The work of Albreshne *et al.* [19] is towards developing a 'semi-automatic' system that uses the semantic information of the services to put together a composition that conforms as closely as possible to the articulated requirements. They use the motivating example of a home electrical system with varied appliances. The constraints are the requirements of the dwellers and the amount of power consumed and heat produced. The semantic information of the available electrical services is utilized to satisfy the requirements as far as possible while being within the specified limits.

Mrissa *et al.* [20] deal with compositions that utilize semantic information along with the context in which the composition is used. The semantic information is utilized to make appropriate service selections dynamically and is obtained from the available standard service ontologies. The contextual information on the other hand is gathered through sensors of various kinds and enables making even better selections.

Semantic service composition primarily deals with the selection of appropriate service elements for composition. This is mainly what our approach is also directed towards. There are however a few important elements missing in semantic service composition that our approach seeks to include. First, the semantic service composition techniques mainly utilize semantic information related to the core functionality of the service. They typically do not involve the non-functional attributes of services in the selection process. The rapid proliferation of services has today resulted in the availability of large numbers of service elements of similar functional capabilities. In such a scenario, non-functional attributes serve as discriminating factors in the service selection process. The absence of the use of non-functional attributes in making appropriate service selections in semantic composition techniques is a gap that needs to be investigated. Our technique does this by making the non-functional attributes a major part in the selection decisions along with the functional capabilities of the services. The number and type of non-functional attributes in our technique can vary depending upon the nature of the application and/or the requirements.

The second weakness in semantic service composition techniques is the absence of active customer involvement in the service composition process. Customers may provide some input preferences during the planning phase of the composition but do not have the flexibility to modify their preferences at a later stage in the composition. On the contrary, our approach provides a great deal of flexibility to customers in articulating their preferences and then tuning their preferences exactly to meet evolving needs at later stages in the composition.

2.2.4 Declarative composition

Declarative compositions are based on a declared abstract description. This basically means that the abstract structure of the composition is declared in advance, but the selection of service elements to instantiate the abstract description is done dynamically at run-time. The usual procedure followed in this approach is to utilize optimization techniques to select service elements that are a good balance between the declared structure and constraining requirements such as cost, or time. The optimization techniques are usually from operations research literature. A few useful Operations Research techniques in use in Declarative composition are: Linear Programming, Non-linear Programming, Dynamic Programming, Queueing Theory, etc.

There is significant work done in this direction. VanderMeer *et al.* [21] propose the FUSION system that allows the customer to declare the abstract structure of the service composition. Using the abstract structure, the system dynamically selects service elements corresponding to the specified structure. The system further refers the selected services to the customer for his/her approval. If the customer is not satisfied, new selections are made.

Channa *et al.* [22] discuss a technique for dynamic selection of service elements with a set of business constraints. The constraints are factors such as service completion time, or type of supplier. The constraints are represented as an ontology for the proper incorporation of relations among the constraints. The service selection process with the constraints is next represented as a constraint satisfaction problem [91] and is solved by a Linear Programming technique [43].

The SELF-SERV technique proposed by Benatallah *et al.* [23] comprises representing the abstract declarative specification of the composition using a state-chart. The abstract functionalities in the state-chart representation are instantiated by service "communities". A service community is a registry where service elements are registered based on their functional capabilities. The service community provides an appropriate service for every abstract functionality of the state-chart at run-time and a dynamic service composition is formed.

The service composition approach presented in this thesis falls in the category of declarative compositions. In our approach the abstract functionality is specified in advance and techniques are proposed to dynamically populate the specified functionalities. However, there are two important aspects that lead us to believe that our technique is superior to the ones described. First, our technique incorporates multiple attributes simultaneously in making service selection decisions for composition. Most of the existing techniques incorporate only one attribute, the basic functionality, to make selections. The technique proposed by Channa *et al.* [22] however does claim to incorporate an ontology of business constraints in making service selections. In doing this, they incorporate multiple attributes in the service selection process but the attributes are only ones that are closely related to each other in the constraint ontology. Our technique on the other hand is able to simultaneously incorporate multiple non-functional attributes (that may be totally

unrelated to each other) in addition to the basic functionality of the service in the composition process. Some of the existing declarative composition techniques also use a few non-functional attributes but these are usually secondary features that are encapsulated as constraints such as time, and cost. None of the existing techniques pays central attention to the non-functional attributes like our technique does.

Secondly, the involvement of customers in the service composition process in the existing declarative techniques is mostly peripheral. The customers at best articulate their preferences once at the beginning and have no further role to play in the composition process. The FUSION system does involve customers in the declaration of the abstract structure but the customers have little say in setting the parameters based on which the composition is formed. Our technique gives a primary role to customers in the composition process. Customers need to articulate their preferences on the basis of which the composition is formed. Flexibility is provided for the customers to modify their preferences at any stage in the composition process.

2.3 Our proposed service composition approach

The proposed service composition approach is a declarative composition procedure where the abstract functionality is declared in advance and the selection of service elements to instantiate the functionalities is done at run-time. The selection of the service elements is done on the basis of the service attributes (basic functionality and non-functional attributes) of the service elements and the preferences articulated by the customers. A salient feature of the proposed technique is that it is able to simultaneously incorporate any number of service attributes in the service composition process.

In this section we give an overview of the service composition technique presented in the thesis. We first discuss the pre-composition set up: the computation of parameters, the preference articulation by customers, etc. The pre-composition activities are followed by the actual composition procedure which is done at run-time using the information gathered in the pre-composition phase.

2.3.1 Pre-composition activities

The pre-composition activities in our service composition approach can be divided into two parts: 1) assessment of the service attributes in the available service elements, and 2) preference articulation by customers. We use a simple trip-planner service example shown in Figure 2.3 as the motivating scenario for demonstrating our technique. The trip-planner application consists of three functionalities: 1) a flight service, 2) a taxi service, and 3) a hotel service. Ours being a declarative dynamic composition approach, the order of invocation of the functionalities is determined in advance. The selection of service elements that provide these functionalities is done dynamically at run-time. Figure 2.3 shows the set of service elements

available from which the selection of the service elements is undertaken.

Figure 2.3: The trip-planner scenario

Assessment of service attributes

The first step in the pre-composition phase of the presented composition approach is assessment of service attributes. The service attributes comprise the basic functionality that the service element is expected to perform and the non-functional attributes. Each service element in the domain is examined separately and the respective service attribute values for the service element are computed or gathered. The computed values of the service attributes are stored in a repository that provides quick and easy access. This is shown in Figure 2.4.

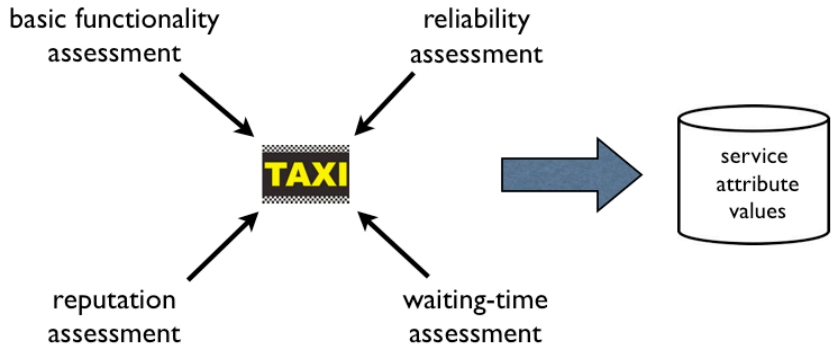


Figure 2.4: Assessment of service attributes in individual service elements

The basic functionality of the service elements differs from one application to another. The basic functionality can include attributes such as number of passengers that can be accommodated for a taxi service. In this thesis, we assume that the

basic functionality values are available to us. These values could be obtained from information repositories or service registries. We do not examine the procedures of obtaining this data.

The non-functional attributes are usually the primary discriminating factors while making service selection decisions. The non-functional attributes are individually assessed for each service element. Any number or type of non-functional attributes can be chosen as factors for determining service selection. The number and type of non-functional attributes chosen is usually dependent on the nature of the application. In this thesis for demonstration purposes, we choose three non-functional attributes: reliability, waiting-time and reputation. We present novel techniques for assessing these attributes in a service composition environment. Chapter 3 comprises a detailed discussion of our reliability assessment technique and its use in individually selecting service elements. Chapter 4 describes our approach to waiting-time assessment. We present two approaches to waiting-time assessment: i) a static approach which involves computing the anticipated waiting-time values in advance and storing these in a repository as described earlier; and ii) a dynamic approach in which the waiting-time is computed at run-time on the basis of the number of requests waiting at the service element at the time of invocation. The dynamic technique is more precise although much more computationally expensive. Finally, Chapter 5 describes in detail our technique for reputation assessment of service elements based on customer feedback.

The techniques presented for the assessment of the non-functional attributes are shown to be effective and individually validated in the respective chapters. These techniques however can be substituted by or supplemented with other suitable techniques for the assessment of the attributes without affecting the service composition process.

Preference articulation by customers

The next step in the pre-composition phase of the composition approach is the articulation of preferences by the customers. The preference articulation by the customer entails the customer expressing which service attributes he/she prefers to incorporate in the composition process, and the emphasis that should be placed on each service attribute. The preferences articulated by the customers are stored in a repository that can be easily accessed. The preference articulation step is shown in Figure 2.5.

Although the preference articulation is mentioned under "pre-composition activities", preference articulation by customers in our approach is an ongoing activity and customers can modify their preferences at any stage in the composition process and all subsequent selections will be on the basis of the new preferences.

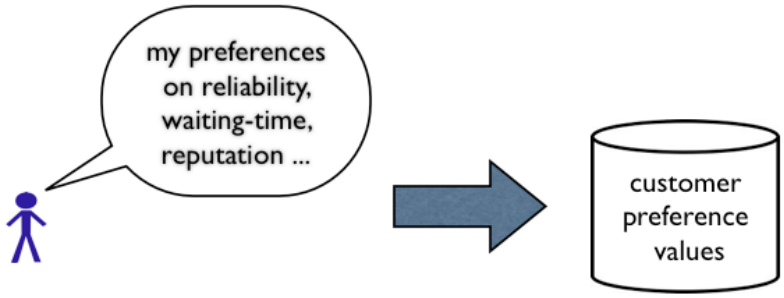


Figure 2.5: Articulation of customer preferences

2.3.2 The composition procedure

The actual service composition procedure is carried out after the pre-composition activities are completed. The individual service attribute values and the customer preferences obtained from the pre-composition phase are combined together through a unifying factor called *affinity*. The affinity factor and its components will be discussed in detail in Chapter 6 of the thesis.

The composition process is demonstrated using the trip-planner example. We assume that the service selection for the flight service is done (*e.g. Air-Canada* is selected) and the next selection to be done is from the Taxi service functionality. This is shown in Figure 2.6.

Figure 2.6: Service selection at the Taxi functionality

The service attribute values and the customer preference values are accessed from the repository and the unifying factor affinity is computed for every potential taxi service as shown in Figure 2.7. The calculation of the affinity factor is done

at run-time. The service element which has the largest value of affinity is the one selected.

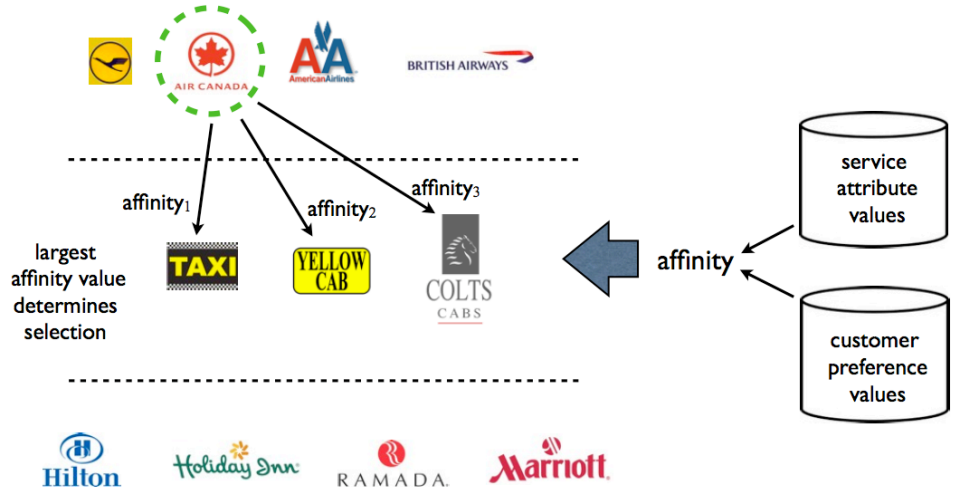


Figure 2.7: Service selection at the Taxi level based on affinity

This simple procedure is followed for all subsequent functionalities in the domain. At each functionality a service element is dynamically selected and in this way the service composition, {*Air Canada*, *Yellow Cab*, *Holiday Inn*}, is formed as shown in Figure 2.8.

Figure 2.8: Composition formed through selection at each level

2.3.3 Assumptions

The proposed service composition model makes a number of simplifying assumptions about the composition process and the environment in which the composition is carried out. These are enumerated below:

1. The service element instances that populate the domain are assumed to be already available. We do not explore the issue of service discovery. Service discovery is extensively discussed in [63].

2. The underlying technology used in invoking service elements is not discussed in this thesis. The composition process is described at a higher level of abstraction.

3. The composition of the service domain is dynamic. Service element instances are assumed to enter and leave the domain continuously.

4. Relevant data required for service attribute computation is assumed to be readily available.

5. Governance and legal issues are assumed to be absent in spite of the inter-organizational nature of the composition environment.

We do not discuss the consequences of making these assumptions in this thesis. A few of these assumptions raise interesting questions which we hope to explore in future. Specifically the assumption on governance in a service composition environment is an interesting and important issue and would make an ideal topic for further research.

2.4 Summary

This chapter sets the stage for a more detailed description of the procedures and techniques presented in this thesis. We first briefly discussed services and service attributes. The discussion on service attributes is imperative because they play a central role in the service composition approach presented in the thesis. The related work section on dynamic service compositions was next and gave us a perspective on the existing approaches to dynamic service composition and where our approach fits. Finally, Section 2.3 described the sequence of steps to be followed in our approach and established the road-map for the remainder of the thesis.

Chapter 3

Reliability

In this chapter[1] and in Chapters 4 and 5, we present novel techniques to calculate the values of individual non-functional attributes of services in a composition environment. The non-functional attributes that we consider are: *reliability, waiting-time,* and *reputation.*

A good definition of reliability is: *"... reliability has been defined as the probability that a ... fault which causes deviation from required output by more than specified tolerances, in a specified environment, does not occur during a specified exposure period"* [38]. This definition is in the context of software but holds for any system in general. Reliability is arguably the most important non-functional attribute of a service from the customer point of view [73] – if services fail often, they are of little use to customers. Numerous techniques for the assessment of reliability of services are available [58] [59] [60] [13]. Most of these techniques, we feel, are not suitable in a composition environment, because they consider each service element in isolation while computing the respective reliability measures. This is inappropriate because in a composition environment there are interactions between service elements. This interaction is bound to affect the reliability of individual service elements. In this chapter, we propose a technique that takes into account the interactions between service elements while calculating the reliability values of individual service elements. The technique comprises the representation of the service domain as a Continuous Time Markov Chain (CTMC) which takes into account the interactions between service elements. The CTMC representation is then utilized to assess the reliability values of individual service elements.

3.1 Related work

Substantial research exists on computing the reliability of a composite application on the basis of the individual reliabilities of its constituent service elements. The

[1] A first version of this chapter has been published in the Proceedings of the Sixth International Conference on Web Information Systems and Technologies (WEBIST) [98] and an extended version has been published in the Lecture Notes in Business Information Processing (LNBIP) [102]

technique discussed by Yu *et al.* [58] is for Quality of Service (QoS) factors in general and reliability is one among a number of factors. Their approach is quite basic. They assume that the reliability values of individual service elements are given and the reliability of the composite application is simply the product of the reliabilities of individual constituent service elements.

Tsai *et al.* [59] present a method of first calculating the reliability of individual service elements using a technique called *group testing*. In this technique, the service element in question is put through the same test as a number of other functionally equivalent service elements of known reliabilities. The other service elements are assigned appropriate weights on the basis of their reliability values. Depending on how close or far the result of the service element in question is from that of the other services, the service element is assigned a weight which is an approximate reflection of its reliability. Next, using the reliability of the individual service elements so calculated, the reliability of the composite application is calculated using one of various simple formulae depending on how the individual service elements are connected in the application: as a *sequence, choice, loop* or *concurrently.*

Zeng *et al.* [13] represent the various possible composite applications as a Directed Acyclic Graph (DAG). In this DAG, they find a 'critical path'. The reliability of the composite application is determined by the expression $e^{rel(s_i)*z_i}$ where $rel(s_i)$ is the reliability of a service element i. $z_i = 1$ if the service element i falls in the critical path, and $z_i = 0$ otherwise. The reliabilities of individual service elements are calculated using historical data as the ratio of successful service delivery to total number of invocations.

Wang *et al.* [60] present an interesting technique for calculating the reliability of the service composition. Their method assumes that the transition behaviour of the system from one service element to another follows a Markov process [50]. They express a reliable transfer from each service element to every other service element in the form of a matrix. The reliability of a service composition is expressed by assigning an exponent to this matrix that is equal to the number of transition steps required to move from the initial service element to the final service element of the composition. Subsequently, the matrix is assigned exponents of all possible integer values from 0 to ∞, representing all possible service compositions and the summation of this geometric progression gives the reliability of the composition.

Epifani *et al.* [62] present a dynamic modeling technique wherein Bayesian estimation is used to dynamically revise the non-functional attribute values of the constituent service elements of a composite application. The posterior distribution of the values of the non-functional attributes such as reliability are determined on the basis of prior distribution and a better service composition is achieved.

All these methods, depend on the assumption that the reliability of individual service elements in a service domain can be calculated in isolation. In reality though, in an environment like this where service elements interact in a complementary manner to form composite applications, coupling among service elements does exist. This coupling arises out of business relationships, and ease of inter-service

usage capabilities between service elements. In this thesis, we assign a value to the coupling between interacting service elements. The higher this value, the greater is the coupling. The reliability calculation of individual service elements is therefore done incorporating these coupling values. Subsequently, the service elements with the best individual values of reliability, so calculated, collectively contribute to the best reliability values for the composite application.

3.2 Service domain representation

The service domain comprises a set of service elements providing different types of functionalities. Dynamic composition involves making selections from the available service elements at run-time and tying together the service elements to form the composition. An example of a service composition is a financial service. The functionalities of such a service set could be: a) opening an account, b) applying for a credit card, c) authenticity verification through a digital signature, etc.

The representation of the service domain in our work is done as a *multi-tier acyclic graph*, where each tier represents a certain functionality and is populated by the set of service elements that possess the capability to perform the respective function. The size of this set is dynamic in nature and is characterized by frequent entry and exit of service elements. The left panel of Figure 3.1 shows a simple representation of this. Service elements B and C, which are at the same level, represent functionally equivalent service elements. The same holds for service elements D, E, F, as well as G, H. The work-flow is from top to bottom. A service element at a certain level 'invokes' one of the service elements immediately below it, and subsequently this invoked service element invokes one of the service elements from the next level, and so on. In this way, a service composition is formed. An example of this service domain representation (e.g., a financial service) is shown on the right panel of Figure 3.1.

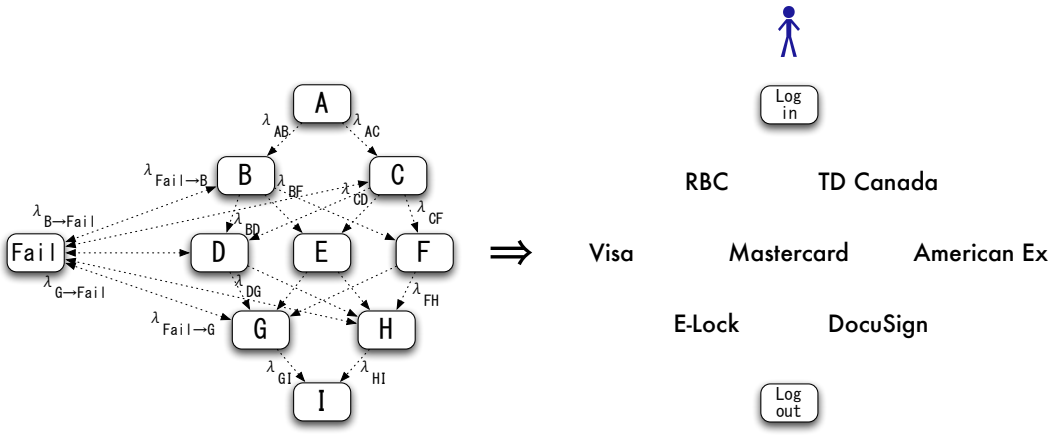


Figure 3.1: The service domain

A variable called *coupling* is assigned between a service element in the domain and every service element at the functionality immediately after (below) it. The coupling C_{ij}, depends on factors such as business relationships between the providers of services i, and j and ease of inter-service usage of the two services. A high value of C_{ij} indicates a high degree of coupling between i, and j. For example, suppose the banking service *RBC* has an agreement with *Mastercard* wherein customers who open an account with RBC are given incentives to apply for a Mastercard credit card rather than Visa or American Express. This would translate into a high value of coupling between RBC and Mastercard. The values of coupling between service elements range from 1 to ∞. A value of 1 indicates ordinary interaction with no special bonding between the services, whereas higher values indicate better bonding. Different techniques can be employed to compute coupling values. We do not look into the issue of computing coupling values in this thesis.

For each service element in the domain, there is a service completion rate which is the rate at which the service element completes the requests. The service completion rate of service element i, μ_i is calculated as follows:

$$\mu_i = \frac{\textit{no. of service requests completed in a given time period}}{\textit{time period}} \tag{3.1}$$

Subsequently, a transition rate (λ_{ij}) between a calling service element i, and the called service element j is calculated as:

$$\lambda_{ij} = C_{ij} \cdot \mu_i \tag{3.2}$$

where C_{ij} is the coupling between service elements i, and j, and μ_i is the service completion rate of service element i. The transition rates are indicated as labels on the dotted arrows between services in the left panel of Figure 3.1.

Besides these transition rate values, there is a fail-rate ($\lambda_{i \to fail}$) 'arrow' from each service element i in the domain to the fail state. The independent fail-rate of each service element i is expressed as follows,

$$\lambda_{i \to fail} = \frac{\textit{no. of times service i fails}}{\textit{total up-time}} \tag{3.3}$$

There is correspondingly a recovery-rate ($\lambda_{fail \to i}$) from the fail state to each service element i in the domain. The recovery-rate is calculated as follows,

$$\lambda_{fail \to i} = \frac{\textit{no. of times service i recovers from failure}}{\textit{total down-time}} \tag{3.4}$$

The behavior of service elements at each level in the domain is assumed to be unaffected by the service selections prior to it in the domain, *i.e.* by the service selections at the functionalities above it. The service domain can thus be represented as a Continuous Time Markov Chain (CTMC) [50]. This is done by first representing the service domain as a *Stochastic Petri-Net* [51], and then converting the same to the equivalent CTMC representation. A detailed description of the conversion of

the representation of a system to its equivalent CTMC representation is given in a paper on Stochastic Petri Nets by Balbo [52].

The motive behind representing the service domain as a CTMC is to better analyze the system, and come up with useful techniques to assess the reliability of the system.

	Log in/out	RBC	TD Canada	Visa	Mastercard	American Ex	E-Lock	DocuSign
Log in/out	--	1.0	1.0	--	--	--	--	--
RBC	--	--	--	1.4	1.7	1.5	--	--
TD Canada	--	--	--	1.1	2.1	1.9	--	--
Visa	--	--	--	--	--	--	1.4	2.7
Mastercard	--	--	--	--	--	--	2.2	1.9
American Ex	--	--	--	--	--	--	1.2	1.7
E-Lock	1.0	--	--	--	--	--	--	--
DocuSign	1.0	--	--	--	--	--	--	--

(a)

	Failure rate	Recovery rate	Completion rate
Log in/Log out	--	--	1.0
RBC	0.2	0.7	1.7
TD Canada	0.08	0.5	2.4
Visa	0.05	0.009	1.4
Mastercard	0.3	0.002	3.2
American Ex	0.08	1.1	1.8
E-Lock	0.2	0.06	0.8
DocuSign	0.3	0.7	1.4

(b)

Figure 3.2: (a) Coupling values, (b) Initial failure, recovery, and service completion rate values

The transition rates between states in the CTMC are expressed in the form of a matrix called the *infinitesimal generator matrix* (IGM). The value of the ij^{th} element of this matrix is equal to the value of the rate of transition from the i^{th} state to the j^{th} state of the CTMC. As a rule in an IGM, the sum of the elements in each row of the matrix is always zero. Therefore the ii^{th} element is made the negative of the summation of the other elements in the i^{th} row.

The transition rates in the case of our service domain are the transition-rate values between the states representing the usage of service elements (hereafter referred to in this chapter as 'service element state') calculated using equation (3.2), and the transition-rate values between the service element state and the fail state calculated using equations (3.3) and (3.4). The infinitesimal generator matrix for the financial service example whose coupling values between service elements and service completion rate values are shown in Figures 3.2 (a) and 3.2 (b) is given in Figure 3.3.

For example, the transition-rate value from service element state *Mastercard* to *DocuSign* is calculated using equation (3.2) as follows (shown in bold in the matrix):

$$\begin{aligned}\lambda &= Coupling_{Mastercard\text{-}DocuSign} * Completion\text{-}rate_{Mastercard} \\ &= 1.9 * 3.2 = 6.08\end{aligned}$$

The last column and row ($9th$ from left and top) of the matrix correspond to the

$$IGM = \begin{bmatrix} -2.0 & 1.0 & 1.0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -8.02 & 0 & 2.38 & 2.89 & 2.55 & 0 & 0 & 0.2 \\ 0 & 0 & -12.32 & 2.64 & 5.04 & 4.56 & 0 & 0 & 0.08 \\ 0 & 0 & 0 & -5.79 & 0 & 0 & 1.96 & 3.78 & 0.05 \\ 0 & 0 & 0 & 0 & -13.42 & 0 & 7.04 & \mathbf{6.08} & 0.3 \\ 0 & 0 & 0 & 0 & 0 & -5.3 & 2.16 & 3.06 & 0.08 \\ 0.8 & 0 & 0 & 0 & 0 & 0 & -1.0 & 0 & 0.2 \\ 1.4 & 0 & 0 & 0 & 0 & 0 & 0 & -1.7 & 0.3 \\ 0 & 0.7 & 0.5 & 0.009 & 0.002 & 1.1 & 0.06 & 0.7 & -3.071 \end{bmatrix}$$

Figure 3.3: Infinitesimal Generator Matrix

fail-state, and its elements are simply obtained from the table in Figure 3.2 (b) as the initial fail-rate and recovery-rate values respectively of each service element.

A number of analysis techniques require that the transition values between the different states of a CTMC be expressed as a probability. To take advantage of these techniques, an approximate transition probability matrix is obtained from the infinitesimal generator matrix, which is called the *embedded Markov chain matrix*. Each ij^{th} element of this matrix is obtained as the ratio of the corresponding element of the infinitesimal generator matrix to the sum of all the elements except the ii^{th} element in the corresponding row of the infinitesimal generator matrix. The ii^{th} element of the embedded Markov chain matrix, as a rule, is always zero. The embedded Markov chain matrix P for the financial service example corresponding to the infinitesimal generator matrix is shown in Figure 3.4.

$$P = \begin{bmatrix} 0 & 0.5 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.297 & 0.36 & 0.318 & 0 & 0 & 0.025 \\ 0 & 0 & 0 & 0.214 & 0.409 & 0.37 & 0 & 0 & 0.007 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.339 & 0.653 & 0.008 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.525 & \mathbf{0.453} & 0.022 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.408 & 0.577 & 0.015 \\ 0.8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.2 \\ 0.824 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.176 \\ 0 & 0.228 & 0.163 & 0.003 & 0.001 & 0.357 & 0.02 & 0.228 & 0 \end{bmatrix}$$

Figure 3.4: Embedded Markov Chain Matrix

As an example, the element in the embedded Markov chain matrix corresponding to the element in bold in the IGM is calculated as follows:

$$P = \frac{6.08}{7.04 + 6.08 + 0.3} = 0.453$$

3.3 Reliability in terms of 'failure distance'

The CTMC representation of the service domain is utilized to calculate the reliability values. More precisely, the CTMC representation is used to find the 'failure distance' of individual service element states, and this failure distance is regarded as an expression of reliability of the corresponding service element. The larger the failure distance of a service element state, the higher is the reliability of the service element.

The failure distance of a service element state is measured as the 'number of transitions' that the system goes through before converging to the 'fail' state, given that it starts at the service element state in question. The larger the number of transitions, larger the failure distance. This is further elaborated upon in the subsequent portion.

The main component of the CTMC representation of the service domain is the infinitesimal generator matrix (shown in Figure 3.3 for our example) which consists of the transition rate values between every pair of 'invoking' and 'invoked' service element states in the domain. The other important component of the CTMC representation is the *probability vector*. The elements in the probability vector correspond to the various states of the system. Each element in the vector represents the probability that the system exists in the corresponding state at a certain stage. Initially, when number of transitions, $t = 0$, the probability vector is represented as π_0, where

$$\pi_0 = \{p_1^0, p_2^0, p_3^0, \ldots, p_n^0\} \tag{3.5}$$

$p_i^0 (i = 1, 2, \ldots, n)$ are the values representing the probability that the system initially is in state i. Subsequently, after the first transition, $t = 1$,

$$\pi_1 = \pi_0 \cdot P \tag{3.6}$$

where P is the embedded Markov chain matrix (shown in Figure 3.4 for our example). As mentioned earlier, the embedded Markov chain matrix consists of the transition probability values from a state in the system to every other state. It is analogous to the transition probability matrix of a discrete time Markov chain [50].

If we continue multiplying the probability vector with the embedded Markov chain matrix P, we will eventually arrive at a probability vector, π_s, which remains constant, *i.e.* it does not change with further multiplication with P.

$$\pi_1 = \pi_0 \cdot P \Rightarrow \pi_2 = \pi_1 \cdot P \Rightarrow \cdots \pi_s = \pi_s \cdot P \tag{3.7}$$

π_s is called the *equilibrium probability vector*. Each element in this vector represents the probability that the system is in the corresponding state at equilibrium.

We now show that the equilibrium probability vector is the same as the left-hand eigenvector of matrix P corresponding to the unit eigenvalue [55]. If e_i^T is the left-hand eigenvector of a matrix P corresponding to the eigenvalue λ_i, then we know that:

$$\lambda_i \cdot e_i^T = e_i^T \cdot P \tag{3.8}$$

when $\lambda_i = 1$, *i.e.* the unit eigenvalue, then

$$e_i^T = e_i^T \cdot P \tag{3.9}$$

Observing equations (3.7) and (3.9) together,

$$e_i^T = e_i^T \cdot P \Leftrightarrow \pi_s = \pi_s \cdot P \Rightarrow e_i^T = \pi_s$$

The equilibrium probability vector thus is easily calculated as the left-hand eigenvector (normalized to sum to 1) corresponding to the unit eigenvalue of the embedded Markov chain matrix of any service domain. The equilibrium probability vector for the financial service example is shown below:

$$[0.2092, 0.1173, 0.1137, 0.0593, 0.0888, 0.0991, \ldots$$

$$\ldots 0.1083, 0.1488, \mathbf{0.0555}] \tag{3.10}$$

Of the elements of this equilibrium probability vector, the one that corresponds to the fail state (shown in bold) of the CTMC representation, gives the probability that the system is in the fail state at equilibrium. If the probability of the system being in the fail state at equilibrium is high, we conclude that the service element state that is far from equilibrium is also far from the fail state and vice-versa. Therefore, if we somehow find a way of calculating the distance of a service element state from equilibrium, it would also give us an estimate of its distance from failure. Thus the problem of finding the distance of a service element state from failure is translated to that of finding the distance of the service element state from equilibrium.

We utilize the method put forward by William J. Stewart to calculate the distance of the service element state from equilibrium at the various stages of service selection [56]. The method is explained as follows:

Assuming x_1^0 represents the probability vector that models a system that has a 100% probability of being in state 1 initially when the number of transitions $t = 0$,

$$x_1^0 = \{1, 0, 0, \ldots, 0\} \tag{3.11}$$

similarly,

$$x_2^0 = \{0, 1, 0, \ldots, 0\} \tag{3.12}$$

and in general,

$$x_i^0 = \{0, 0, 0, \ldots, 0, 1, 0, \ldots, 0\} \tag{3.13}$$

Let the left-hand eigenvectors of matrix P be

$$\{e_1^T, e_2^T, \ldots, e_n^T\}$$

corresponding respectively to the eigenvalues,

$$\{\lambda_1, \lambda_2, \ldots, \lambda_n\}$$

Since x_i^0 in equation (3.13) is a row vector, it can be expressed as a linear combination of other row vectors. Thus,

$$x_i^0 = c_{i1} \cdot e_1^T + c_{i2} \cdot e_2^T + \ldots + c_{in} \cdot e_n^T \tag{3.14}$$

where $c_{ij}(j = 1, 2, \ldots, n)$ are currently unknown constants whose values and significance are subsequently discussed.

Just like π_1 was computed in equation (3.6), x_i^1 which is the probability vector after the first transition $t = 1$ given that the system starts (at $t = 0$) with a 100% probability of being in state i, is also computed as,

$$\text{equation (3.6): } \pi_1 = \pi_0 \cdot P \Rightarrow x_i^1 = x_i^0 \cdot P$$

Thus, from equation (3.14), it follows that,

$$x_i^1 = x_i^0 \cdot P = c_{i1} \cdot e_1^T \cdot P + c_{i2} \cdot e_2^T \cdot P + \ldots + c_{in} \cdot e_n^T \cdot P \tag{3.15}$$

Using equation (3.8): $\lambda_i \cdot e_i^T = e_i^T \cdot P$, we get,

$$x_i^1 = c_{i1} \cdot \lambda_1 \cdot e_1^T + c_{i2} \cdot \lambda_2 \cdot e_2^T + \ldots + c_{in} \cdot \lambda_n \cdot e_n^T \tag{3.16}$$

Similarly, we may get x_i^2 as,

$$\begin{aligned} x_i^2 &= x_i^1 \cdot P \\ &= c_{i1} \cdot \lambda_1 \cdot e_1^T \cdot P + c_{i2} \cdot \lambda_2 \cdot e_2^T \cdot P + \ldots + c_{in} \cdot \lambda_n \cdot e_n^T \cdot P \\ &= c_{i1} \cdot \lambda_1^2 \cdot e_1^T + c_{i2} \cdot \lambda_2^2 \cdot e_2^T + \ldots + c_{in} \cdot \lambda_n^2 \cdot e_n^T \end{aligned}$$

After a certain number of transitions, the vector x_i eventually converges to the equilibrium probability vector. Say after k steps,

$$x_i^k = c_{i1} \cdot \lambda_1^k \cdot e_1^T + c_{i2} \cdot \lambda_2^k \cdot e_2^T + \ldots + c_{in} \cdot \lambda_n^k \cdot e_n^T \tag{3.17}$$

Now similarly, assuming x_j^0 represent the probability vector such that there is a 100% probability of the system being in state j initially when the number of transitions $t = 0$. Thus,

$$x_j^0 = \{0, 0, 0, \ldots, 0, 1, 0, \ldots, 0\}$$

Just like x_i^0 in equation (3.14), x_j^0 is also expressed as a linear combination of the left hand eigenvectors of P,

$$x_j^0 = c_{j1} \cdot e_1^T + c_{j2} \cdot e_2^T + \ldots + c_{jn} \cdot e_n^T$$

Similarly,

$$\begin{aligned} x_j^1 &= c_{j1} \cdot e_1^T \cdot P + c_{j2} \cdot e_2^T \cdot P + \ldots + c_{jn} \cdot e_n^T \cdot P \\ &= c_{j1} \cdot \lambda_1 \cdot e_1^T + c_{j2} \cdot \lambda_2 \cdot e_2^T + \ldots + c_{jn} \cdot \lambda_n \cdot e_n^T \end{aligned}$$

and after k steps,

$$x_j^k = c_{j1} \cdot \lambda_1^k \cdot e_1^T + c_{j2} \cdot \lambda_2^k \cdot e_2^T + \ldots + c_{jn} \cdot \lambda_n^k \cdot e_n^T \tag{3.18}$$

Observing equations (3.17) and (3.18) together, the only difference between the two equations are the constants c_{il}, and c_{jl} where $l = 1, 2, \ldots, n$. In other words, the difference in the state of the system after k transition steps, when the starting state is i, and when the starting state is j, is represented by the difference in the values of the constants c_{il}, and c_{jl}, where $l = 1, 2, \ldots, n$. The values of c_{il}, and c_{jl}, ($l = 1, 2, \ldots, n$) therefore hold the key to finding the difference in the distance (in terms of number of transition steps) of state i, and state j from any other state of the system. Thus, these constants also reflect the difference in distance of the two states from the equilibrium state.

The following enables the calculation of $c_{1l}, c_{2l}, c_{3l}, \ldots, c_{nl}$, where $l = 1, 2, \ldots, n$:

$$\begin{aligned} x_1^0 &= \{1, 0, 0, \ldots, 0\} \\ x_2^0 &= \{0, 1, 0, \ldots, 0\} \\ &\vdots \\ x_n^0 &= \{0, 0, 0, \ldots, 1\} \end{aligned}$$

Writing this in matrix form,

$$X = \begin{bmatrix} 1 & 0 & 0 & \ldots & 0 \\ 0 & 1 & 0 & \ldots & 0 \\ 0 & 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & 1 \end{bmatrix} = \text{Identity Matrix} \tag{3.19}$$

We also know that,

$$\begin{aligned} x_1^0 &= c_{11} \cdot e_1^T + c_{12} \cdot e_2^T + \ldots + c_{1n} \cdot e_n^T \\ x_2^0 &= c_{21} \cdot e_1^T + c_{22} \cdot e_2^T + \ldots + c_{2n} \cdot e_n^T \\ &\vdots \\ x_n^0 &= c_{n1} \cdot e_1^T + c_{n2} \cdot e_2^T + \ldots + c_{nn} \cdot e_n^T \end{aligned}$$

Writing this in matrix form as well,

$$\begin{bmatrix} x_1^0 \\ x_2^0 \\ x_3^0 \\ \vdots \\ x_n^0 \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & c_{13} & \ldots & c_{1n} \\ c_{21} & c_{22} & c_{23} & \ldots & c_{2n} \\ c_{31} & c_{32} & c_{33} & \ldots & c_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_{n1} & c_{n2} & c_{n3} & \ldots & c_{nn} \end{bmatrix} \begin{bmatrix} e_1^T \\ e_2^T \\ e_3^T \\ \vdots \\ e_n^T \end{bmatrix} \tag{3.20}$$

Therefore,

$$X = CE^T$$

From equation (3.19), we know that X is the identity matrix (I), therefore,

$$I = CE^T \Rightarrow C = \frac{I}{E^T} = (E^T)^{-1}$$

$(E^T)^{-1}$ is the matrix of the right-hand eigenvectors of matrix P. The values of the constants c_{ij}, that are capable of determining the distance of the various states from equilibrium, are easily computed as the right-hand eigenvectors of matrix P. Of these right-hand eigenvectors, the sub-dominant eigenvector gives the *best* estimate of the distance of each state from equilibrium [56]. The sub-dominant eigenvector for our financial service example is shown below:

$$[0.3749, 0.3566, 0.3606, 0.3513, \mathbf{0.3435}, 0.3476, \ldots$$

$$\ldots 0.3352, 0.3420, 0.1038] \tag{3.21}$$

Translating these results to the service domain, we are in a position to calculate via the CTMC representation of the domain, the 'distance' of each service element from equilibrium.

The important point here, however, is that the distance that needs to be calculated is the failure distance and not necessarily the equilibrium distance. A clear understanding on the relationship between the failure state and the equilibrium state needs to be established. A simple example helps in understanding this.

Suppose, the failure probability at equilibrium is very high (say 0.9). This means that the probability that the system at equilibrium is in the fail state is 0.9. This is represented by the element corresponding to the fail state in the equilibrium probability vector π_s of equation (3.7). In such a scenario, the larger the distance of a service element state from equilibrium, larger the failure distance and hence higher the reliability of the service element. Conversely, suppose the failure probability at equilibrium is a small value (*e.g.* 0.2). In this case, the smaller the distance of a service element state from equilibrium, higher the reliability of the service element. In general, if the failure probability at equilibrium is greater than 0.5, a larger distance from equilibrium reflects higher reliability and vice-versa for a failure probability smaller than 0.5.

The magnitude of the i^{th} element of the subdominant right-hand eigenvector of the embedded Markov chain matrix (P) of the CTMC representation of the service domain, c_{i2}, gives the *best* estimate of the distance of the i^{th} state from equilibrium, as mentioned earlier [56]. The larger the value of c_{i2}, larger the distance of the i^{th} state from equilibrium. As noted above, a failure probability larger than 0.5 at equilibrium would warrant a larger distance from equilibrium as more reliable and a failure probability at equilibrium less than 0.5 makes a smaller distance from equilibrium more reliable. Mathematically if we have an expression that places c_{i2} in the numerator when the failure probability at equilibrium is greater than 0.5 and

in the denominator when it is less than 0.5, this expression would serve as one for the Failure distance of a service element state. Equation (3.22) is an expression that does this.

$$\textit{Failure Distance} = \frac{1}{\lambda_{i \to Fail} * c_{i2}^{10 \cdot (0.5 - \pi_s(Fail))}} \tag{3.22}$$

In equation (3.22), as long as the failure probability is less than 0.5, the factor $c_{i2}^{10 \cdot (0.5 - \pi_s(Fail)}$ is in the denominator of the expression (since $(0.5 - \pi_s(Fail))$ would be a positive value). Thus, a larger value of c_{i2} (which expresses a larger distance from equilibrium) results in a smaller value of failure distance. Also, the smaller the value of $\pi_s(Fail)$, larger the exponent of c_{i2} and hence the failure distance is smaller. The opposite holds whenever $\pi_s(Fail)$ is greater than 0.5. When the failure probability at equilibrium is exacly 0.5, the factor $c_{i2}^{10 \cdot (0.5 - \pi_s(Fail)}$ is equal to 1 and the failure distance depends only on the current value of the factor $\lambda_{i \to Fail}$. The failure distance values calculated for the state representing the usage of the service elements in our financial service example are: *CIBC* $\rightarrow$ 489.24 ; *TD Canada* $\rightarrow$ 1163.94 ; *Visa* $\rightarrow$ 2091.66 ; *Mastercard* $\rightarrow$ 385.19 ; *American Express* $\rightarrow$ 1370.28 ; *E-Lock* $\rightarrow$ 644.16 ; and *DocuSign* $\rightarrow$ 392.76 . The calculation of failure distance for *Mastercard* is shown below:

$$\begin{aligned} \textit{Failure Distance}_{Mastercard} &= \frac{1}{0.3 * 0.3435^{10 \cdot (0.5 - 0.0555)}} \\ &= 385.19 \end{aligned}$$

The failure distance values so calculated take into account the interactions between the service element states. This is so because the transition rate values between the service element states in the CTMC representation are considered in the calculation.

3.4 Experimental validation

This section describes the experimental validation of the technique proposed to calculate the individual reliabilities of service elements in a domain taking into account the influence of interacting service elements. To do this, experiments were conducted on a service domain with 29 service elements (excluding the first and the last) spread over 6 levels of functionality. Seven different sets of initial fail-rate and coupling values were experimented with. The experimental domain is shown in Figure 3.5.

We ran simulations on the experimental domain, wherein service elements were allowed to fail randomly at their respective fail-rate values. $10,000$ simulation runs were conducted, where each run comprised a failure being forced at each functionality level. Every time a service element failed, its own fail-rate was increased marginally, and so was the fail-rate of the service elements in the functionality level immediately above it. The increase in fail-rate of the parent service elements was

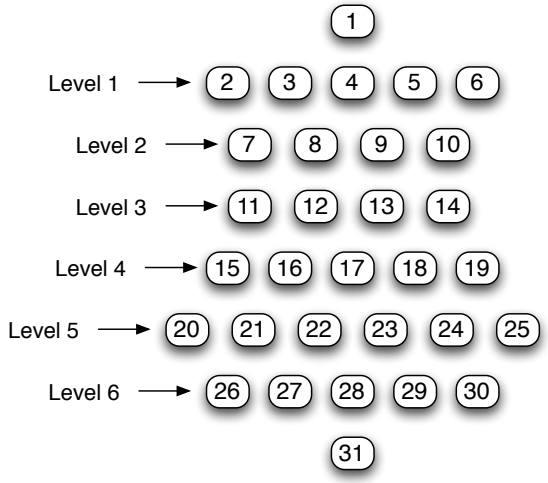


Figure 3.5: Service domain used in the experiments.

proportional to the value of coupling between them and the failing service element. Therefore, a service element with strong coupling with the failing service element suffered a large increase in its fail rate. This was done to conform with the general belief that the reliability value of a service element influences the reliability of the interacting service elements, particularly the ones that invoke it.

A number of simulation scenarios were observed. However, only the two main ones are discussed here. The first simulation scenario was a static scenario, wherein service elements were allowed to fail, their fail-rates on failing were increased, and they were allowed to fail again. The fail-rate of service elements were allowed to rise indefinitely. The second scenario was a more dynamic scenario, wherein service elements were allowed to fail, and their fail-rates were increased on failing as in the static case. However, once the fail-rate of a service element exceeded a certain maximum, the service element was taken-out for 'repair' and brought in after a few simulation runs with its fail-rate reinstated to its original value.

The motivation behind this simulation exercise was to understand the possible system behaviour if it were observed for a very long period of time (one in which $10,000$ failures occurred at each functionality level). The number of failures for each service element returned by the simulation formed the basis for the validation of our proposed technique against the existing techniques for reliability assessment in an interactive environment.

The proposed technique, described in previous sections, involves several steps. These include: 1) constructing the representation of the service domain in its equivalent CTMC form, 2) putting together the 'infinitesimal generator matrix' (shown in Figure 3.3 for the financial service example) of this CTMC, 3) calculating the embedded Markov chain matrix (cf. Figure 3.4 for the financial service example), and 4) subsequently finding the left-hand eigenvector (corresponding to the unit eigenvalue) which is the equilibrium probability vector of the system (cf. equation

Table 3.1: Results of experiments in a static scenario (no repair of services)

Static Scenario				
	Existing Techniques		Proposed Techniques	
	Failures	Reliability	Failures	Reliability
Domain 1	6,050/10,000	39.5%	1,498/10,000	85.92%
Domain 2	8,765/10,000	12.35%	712/10,000	92.88%
Domain 3	9,008/10,000	9.92%	16/10,000	99.84%
Domain 4	9,844/10,000	1.56%	526/10,000	94.74%
Domain 5	9,778/10,000	2.22%	521/10,000	94.79%
Domain 6	6,490/10,000	35.1%	4,236/10,000	57.64%
Domain 7	9,983/10,000	1.7%	170/10,000	98.3%

Table 3.2: Results of experiments in a dynamic scenario (with repair of services)

Dynamic Scenario (repair time: 1 iteration)				
	Existing Techniques		Proposed Techniques	
	Failures	Reliability	Failures	Reliability
Domain 1	8,231/10,000	17.69%	100/10,000	99%
Domain 2	1,168/10,000	88.32%	244/10,000	97.56%
Domain 3	1,969/10,000	80.31%	157/10,000	98.43%
Domain 4	1,420/10,000	85.8%	161/10,000	98.39%
Domain 5	3,705/10,000	62.95%	669/10,000	99.31%
Domain 6	688/10,000	93.12%	181/10,000	98.19%
Domain 7	7,199/10,000	28.01%	4,492/10,000	55.08%

(3.10)) and the subdominant right-hand eigenvector of this matrix. Finally the equilibrium probability vector and the subdominant right-hand eigenvector are used to calculate the 'failure distance' of each service element state in the domain using equation (3.22). The service element selected at each functionality level is the one with the *largest* value of failure distance at that level. This service element is considered to be furthest from the fail state and thus the most reliable.

The existing techniques as discussed in the related work section (Section 3.1) are varied. However, in terms of calculating the individual reliabilities of service elements, all of them are similar in that they all calculate the individual reliabilities of the service elements in isolation. That is, they assume that the reliabilities of individual service elements in the domain are unaffected by their interaction with other service elements in the domain. Therefore, to replicate the existing techniques in this domain, the initial fail-rate values of the service elements are utilized, and at each level the service element which has the *smallest* fail-rate value at that level is selected.

The sequence followed in the experiments for the static scenario is now described. The proposed technique and existing techniques were first applied on the experimental domain. These gave the set of service elements selected at each level. The simulation was next run for $10,000$ iterations each. Each iteration comprised the occurrence of 1 failure at each functionality level. The failing service elements at each functionality level on each iteration were compared with the service elements selected by the two techniques (proposed and existing). If any of the selected service elements by a selection technique happened to fail in an iteration of the simulation, the selection technique was said to be unsuccessful in that iteration. In other words, a selection was a success on a particular iteration if and only if none of the service elements selected (at any level) was one of the failing service elements in the iteration.

The experimental procedure for the dynamic scenario was similar with the only difference being that the selection techniques (both proposed and existing) were applied after every simulation run rather than only once in the beginning. This is because, with service elements being taken out for repair, the complexion of the domain was possibly changing on every run.

Subsequently the reliability of the composite application selected by either selection technique was calculated as shown in equation (3.23).

$$Reliability = \frac{\textit{total no. of iterations} - \textit{total no. of failures}}{\textit{total no. of iterations}} \tag{3.23}$$

The results for the two scenarios are shown in Tables 3.1, and 3.2. The results show that in both cases, the proposed technique outperforms the existing techniques. The results therefore indicate that the proposed technique does manage to compose a more reliable application in a scenario where the service elements are interacting with each other and the reliability of a service element that invokes another is affected by that of the one invoked.

3.5 Summary

This chapter deals with the selection of service elements in a composition scenario making use of their reliability values. A more reliabile service element is selected for composition over a less reliable one. The important contribution of the proposed technique is that it is able to capture the influence that interacting service elements have on the reliability figures of each other. The reliability of individual service elements is expressed as its failure-distance and the same is calculated through the representation of the service domain as a continuous-time Markov chain and utilizing a method proposed by Stewart [56]. The validation of the technique is done by observing a simulation of the service domain for a long period (involving $10,000$ iterations) and assessing the proposed selection technique against the simulation. Existing techniques of reliability calculation which calculate service element reliability treating the elements in isolation are also assessed against the simulation. It

is shown that the proposed technique far outperforms the existing techniques. The composite application thus selected has a higher degree of reliability.

The more general motivation for our work is to form a service composition which has a good balance of the values of a number of non-functional attributes, and not just reliability. The reliability values calculated here can be combined with other attribute values as described in Chapter 6 in an attempt to form a composition that is the best possible in such a scenario.

Chapter 4

Waiting-time

The next non-functional attribute that we examine in this thesis is *waiting-time*. The waiting-time at a service denotes the time that a request has to wait before it *starts* getting served by that service. Waiting time plays an important role in determining the appropriateness of a service element for a composition. A large waiting time at a selected element slows down the entire composition process. As such, service elements with small waiting-time values are preferred.

In our research[1] we developed two approaches for computing the waiting-time for service elements. The first approach is a static approach where the waiting-time at a service element is pre-calculated by analyzing the behavior of the domain over a period of time. Concepts from queueing theory [67] are utilized and appropriately customized to predict the approximate waiting-time at the service elements. The technique is validated experimentally by simulating the service domain.

The other approach to waiting-time calculation is more dynamic. In this approach the waiting-time computation is deferred until the point of service selection. The selection decision is made at run-time based on the immediate distribution of requests in the domain. This approach is more accurate than the static approach although much more resource intensive and expensive. The proposed waiting-time calculation technique is verified mathematically using concepts of mathematical induction [75].

4.1 Related work

The related work included in this section is not necessarily in the context of service composition. However, all the work discussed here is related to the waiting-time attribute being used for service selection and performance modification.

Ismail *et al.* [64] tackle the issue of reducing the waiting-time of a collaborative project by selecting collaborating partners in a manner that the idle time of one coincides with that of the other. They explore patterns which avoid situations where

[1] A version of this chapter has been published in the Proceedings of the Seventh International Conference on Services Computing (SCC) [99].

one partner is occupied with one application while a second partner is idle. The technique is relevant only in a more general situation where the commitments of the collaborating organizations are defined in advance. This technique may not be suitable in a more dynamic environment such as service composition.

Wang *et al.* [65] present a technique that borrows concepts from queueing theory to calculate the expected waiting time of service elements in a composition situation. They make the simplifying assumption that the service composition queue is in the steady-state, by assuming that the request arrival-rate is always smaller than the service completion-rate. This however is not always true. Second, their method of calculating the request arrival-rate is based on observing the behavior of each service element for a long time. This may be misleading if the current behavior is not reflective of previous behavior. This is also usually not practical owing to the dynamism associated with the characteristics of each service element in a composition environment.

Flockhart *et al.* [71] present a technique to minimize the waiting-time for calls in a call-centre set up. The technique involves forming 'agent-queues' for each skill. Whenever a call is received requiring a certain skill, the agents in the queue corresponding to the skill are looked up. The number of queues that each of these agents is present in is computed. The agent that is present in the smallest number of queues is the one selected. This ensures that the waiting-time for subsequent calls is minimized. In a service composition set-up, this technique could be utilized in a situation where each service element is capable of performing more than one task. Whenever a request for a task comes along, the service element that is available, capable of performing the task, and which is capable of performing the minimum number of other tasks amongst all available service elements is selected.

Green [70] utilizes queueing theory in healthcare in the allocation of resources in an effective manner. The complexity and unpredictability of a hospital set-up notwithstanding, Green shows the effectiveness of an $M/M/s$ queueing model in the allocation of resources such as beds and staff to minimize the waiting-time of patients as well as enabling optimal utilization of resources. The demand for flexibility in hospitals is dealt with by the application of the queueing model over smaller 'staffing periods' rather than over the whole day.

4.2 The Static approach

In the static setting, the technique presented for calculating the average waiting-time at each service element involves employing concepts from queueing theory. The queueing theory concepts borrowed, however, are those meant for a 'steady-state' condition when the probability of the queueing system having a certain number of units is fixed [67]. In a service composition environment, unfortunately the steady-state condition is seldom attained. The queueing theory concepts therefore need to be customized to be utilized for the purpose of service composition.

4.2.1 Average waiting-time in queueing theory

The queueing system that is typically utilized is the $M/M/1$ queue. This is the most basic queue, where the first M stands for negative exponential time distribution of request arrivals, the second M stands for negative exponential distribution of service completion, and the 1 stands for the fact that there is just one queue (no parallel queues). Figure 4.1 shows the $M/M/1$ queue where λ denotes the request 'arrival-rate' and μ denotes the service 'completion-rate'.

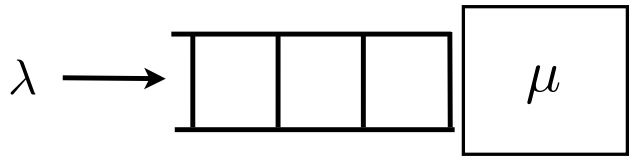


Figure 4.1: The basic $M/M/1$ queue

The $M/M/1$ is the common queue that is observed in most everyday activities like customer queues in banks, movie-halls, etc. A queue like this is characterized by the number of units present within the queueing system. For example, in the queue in a bank, if the number of customers waiting in the queue is 5, the total number of customers in the queueing system is 6 $(= 5+1)$, the 5 waiting customers and the 1 customer being served. The 'state' of that queueing system at that point of time is said to be 6. A queueing system of this kind, for a short period of time has a randomly varying number of states. To characterize the behavior of the queue during short time spans is very challenging. However, if the system is observed over a long period of time, it attains a 'steady-state' wherein the probability of the system existing in any state is constant. In such a situation, the queue becomes more predictable and useful conclusions about the queue can be drawn. To attain the steady-state however, one basic requirement is that the request arrival-rate should be smaller than the service completion-rate of the system ($\lambda < \mu$). If this condition is not satisfied, the number of waiting units progressively increases and the steady-state is never reached.

In the steady-state situation, the average waiting-time for requests in an $M/M/1$ queue is given by the expression in equation (4.1). In this equation, as the value of the arrival-rate λ increases, the value of $(\mu - \lambda)$ in the denominator decreases for a constant completion-rate μ. The increasing λ in the numerator and the decreasing $(\mu - \lambda)$ in the denominator cause the average waiting-time to increase. This makes sense intuitively, as the arrival-rate and hence number of arrivals increases in a queue, the average waiting-time increases. However, this makes sense only as long as λ is less than μ. As soon as the two become equal, according to equation (4.1) the average waiting-time becomes ∞, and if λ increased beyond μ, the average waiting-time becomes negative.

$$\textit{Average waiting time} = \frac{\lambda}{\mu \cdot (\mu - \lambda)} \tag{4.1}$$

4.2.2 Customization of average waiting-time for the composition model

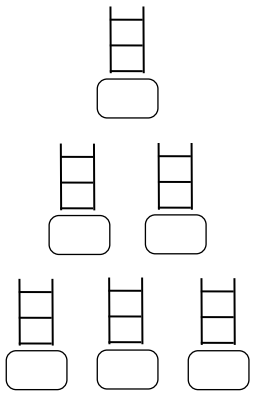


Figure 4.2: Representation of the service domain as a set of $M/M/1$ queues

In our service composition model, the phenomenon of request arrival, waiting of requests at service elements, and finally the servicing of requests at each service element is akin to an $M/M/1$ queue. We therefore represent the service domain of our composition model as a set of $M/M/1$ queues for the purpose of calculating the average waiting time at each service element, as shown in Figure 4.2. For each of these queues the completion-rate is the same as the completion-rate of the service element, however the request arrival-rate is not as straight-forward and needs to be derived from the completion-rate values of all the service elements at the level directly above.

To determine the request arrival-rate for a service element in a multi-level domain, a simple case is first considered. Assume the service domain at each level comprises of only one service element. The request arrival-rate at a service element l would then simply be equal to the completion-rate of the service element k immediately above it. This is because, every request on being served by k would enter the queue of service element l as shown in Figure 4.3.

For a service domain with multiple candidate service elements at each functionality level, a request may arrive from any of the service elements from the level immediately above it. The 'coupling' values described in Section 3.2 are used in determining which candidate service element is chosen by the request. Recall that coupling is a characteristic feature between two service elements i and j at adjacent functionality levels such that a higher coupling value C_{ij} results in a greater likelihood of service element i invoking service j. This means that a request on being serviced by a service element at one level would next move to a service element

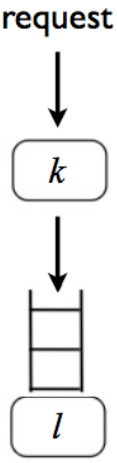


Figure 4.3: Request flow in a domain with a single service element at each functionality

which has a higher value of coupling with the current service element. The request arrival-rate at the queue of the lower service element therefore depends on the value of coupling between the lower service element and the upper service elements and on the completion-rate values of the service levels at the upper level.

Figure 4.4 illustrates this. Service element l is a service element at a certain level which has coupling values C_{il}, C_{jl}, and C_{kl} with services elements i, j, and k respectively at the level directly above it. Service elements i, j, and k respectively have completion-rate values equal to μ_i, μ_j, and μ_k. The value of the request arrival-rate at service element l derived from these completion rate values is shown in equation (4.2).

$$\textit{Arrival-rate}_l(\lambda_l) = \frac{C_{il}}{C_{il}+C_{jl}+C_{kl}} \cdot \mu_i + \frac{C_{jl}}{C_{il}+C_{jl}+C_{kl}} \cdot \mu_j + \frac{C_{kl}}{C_{il}+C_{jl}+C_{kl}} \cdot \mu_k \tag{4.2}$$

The second customization that needs to be done while borrowing the concepts of the $M/M/1$ queueing system is to recognize that the steady state condition is difficult to attain in a dynamic service composition environment. This is due to the fact that it is difficult to ensure that the request arrival-rate at each service element is always less than its completion-rate. Several 'trial and error' experiments were performed to modify the average waiting-time expression of equation (4.1), and finally the average waiting-time expression was modified as shown in equation (4.3) that gives an approximate value of the average waiting-time of a queue in conditions that do not qualify as 'steady-state'. It may be argued that such a modification of the average waiting time expression could deem the latter invalid. We however conducted experiments, results of which are provided in the next section, that establish its validity even in conditions that do not qualify as steady state. Moreover, the modified expression also makes sense intuitively. As the request arrival-rate λ increases for a constant μ, the value of the average waiting-time increases according to equation

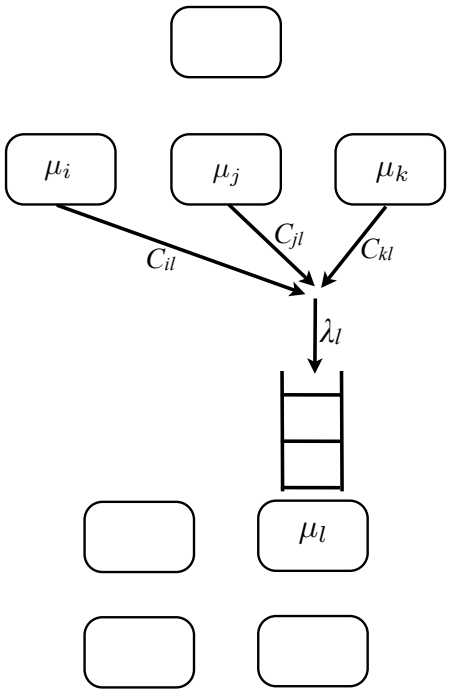


Figure 4.4: The arrival-rate derived from completion rates of ancestor services

(4.3). This sounds logical because an increase in the arrival-rate of requests at a queue for a constant completion-rate would result in a stacking of requests and thus a higher waiting-time. Similarly, according to equation (4.3) as the completion-rate μ increases for a constant λ, the average waiting-time decreases, because the μ is in the denominator of the expression. This also sounds fair because if the completion-rate of the service element increases for a constant arrival-rate, the service element would start servicing the requests much faster and there would be a resulting fall in waiting-time. Furthermore, this expression does not have the requirement of $\lambda < \mu$. It makes intuitive sense even when λ is greater than μ. Thus, the steady-state restriction of the original queueing theory waiting-time expression is overcome.

$$\textit{Average waiting time} = \frac{\lambda}{\mu \cdot (\mu - \lambda)} \Rightarrow \textit{Average waiting time} = \frac{\lambda}{\mu} \tag{4.3}$$

4.2.3 Experimental validation

Experiments were conducted on the hypothetical domain depicted earlier in Figure 3.5. The experimental procedure comprised of simulating the behavior of the domain for a large number of service requests, and calculating the average waiting-time at each service element by observing the time spent by the requests on average in the service queue. Next, the proposed technique was applied on the domain and the waiting-time calculated at each service element. The ranking of the service elements on the basis of the average waiting-time calculated by each technique was found to conform almost exactly at all functionality levels.

Simulation

The simulation procedure used seven different sets of completion-rate values and coupling values on the domain shown in Figure 3.5. Each experiment set also had an application request arrival-rate which was substantially larger (approximately 10 times larger) than any of the completion rate values. $100,000$ events were allowed to happen randomly at the assigned rates. An event comprised either the arrival of a new request for the entire composite application or the completion of any of the individual service element requests.

An application request that arrived at the beginning of the service domain immediately entered service element 1 (Figure 3.5) to be serviced if the latter was idle or joined the queue if service element 1 was busy servicing another request. Once this request was serviced by service element 1, it would move to one of the service elements at the second level. The service to which this request would move to would depend upon two factors: the coupling values between the service elements and service element 1 and the number of requests already waiting in the respective queues of the service elements at the next level. The expression shown in equation (4.4) determined the service element selected next. The selection is illustrated in Figure 4.5. The service element selected is the one which although has a smaller coupling value, has a smaller number of services in its queue.

$$Service\ element\ selected = max(\frac{Coupling_{ij}}{1 + \textit{number of waiting requests}_j}) \tag{4.4}$$

where i is the current service element and j is the service element to be selecte

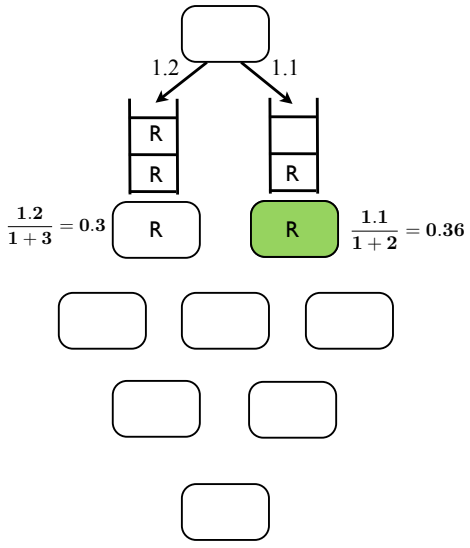


Figure 4.5: The movement of requests from one level to another during simulation

The process of moving from one functionality level to the next continued for the composite service request until it reached the lowest level. This process was

repeated for a large number of requests while noting the time spent by the requests in the queues of each of the service elements. Finally, the average waiting-time for each service element was calculated by dividing the total waiting-time of all the requests that were serviced with the number of requests serviced.

The motivation for running the simulations was to get an idea of how a domain behaved in terms of waiting-time if it were observed for a large period of time (one that involved $100,000$ events).

Results

The proposed technique comprised of first calculating the arrival-rate values for each service element in the domain using the expression in equation (4.2). Subsequently, the calculated arrival-rate values were substituted in the average waiting-time expression in equation (4.3).

The results of the experiments for two sets of completion-rate and coupling values are shown in figures 4.6, and 4.7.

Domain 1		Service element rank (smaller avg. waiting-time ahead)					
Level 1	Simulation	5	2	4	1	3	
	Proposed	5	2	4	1	3	
Level 2	Simulation	4	2	3	1		
	Proposed	4	2	3	1		
Level 3	Simulation	2	3	4	1		
	Proposed	2	3	4	1		
Level 4	Simulation	1	4	5	1	3	
	Proposed	1	4	5	2	3	
Level 5	Simulation	1	4	5	1	5	3
	Proposed	1	4	5	2	6	3
Level 6	Simulation	3	1	5	1	4	
	Proposed	3	2	5	1	4	

Figure 4.6: Level wise comparison of ranking results between simulation and the proposed technique (Domain 1)

The results show the ranks obtained by the service element owing to their respective waiting-time values. The smaller the waiting-time, higher is the rank of the service element. Every pair of rows in the table represents one functionality level of the hypothetical domain of Figure 3.5. For example, the first pair of rows have 5 entries corresponding from left to right the 5 service elements 2, 3, 4, 5, and 6 of the first functionality level of the domain. It can be seen that the ranks obtained by the

service elements through simulation and the proposed technique are almost in perfect conformance at all functionality levels, thus validating the proposed technique.

It should be noted that the values of waiting-time obtained using the customized waiting-time expression in equation (4.3) does not provide correct absolute waiting-time values. Rather, it provides relative waiting-time values so that ranking of the service elements is accurate. This is good enough for the purpose of selecting the 'best' service element in terms of waiting-time.

Domain 2		Service element rank (smaller avg. waiting-time ahead)					
Level 1	Simulation	3	1	5	2	4	
	Proposed	3	1	5	2	4	
Level 2	Simulation	3	1	1	4		
	Proposed	3	2	1	4		
Level 3	Simulation	3	2	4	1		
	Proposed	3	2	4	1		
Level 4	Simulation	2	4	5	3	1	
	Proposed	2	4	5	3	1	
Level 5	Simulation	5	4	2	1	3	5
	Proposed	5	4	2	1	3	6
Level 6	Simulation	4	2	3	5	1	
	Proposed	4	2	3	5	1	

Figure 4.7: Level wise comparison of ranking results between simulation and the proposed technique (Domain 2)

4.3 The dynamic approach

In the technique proposed to calculate the waiting-time dynamically at the service elements, the agent overlooking the service composition has a set of 'satellite' agents at each functionality level of the service domain. The satellite agent at each functionality level returns the current status of each service element at that level in terms of the number of waiting service requests. Figure 4.8 shows a representation of the service domain with the satellite agents.

The service element is not selected until the service request arrives at the given functionality level. When the service request does arrive, the satellite agent provides the current number of requests already waiting in queues at each of the available service elements. This information is then combined with the service completion-rate of the service element and the selection is made using equation (4.5).

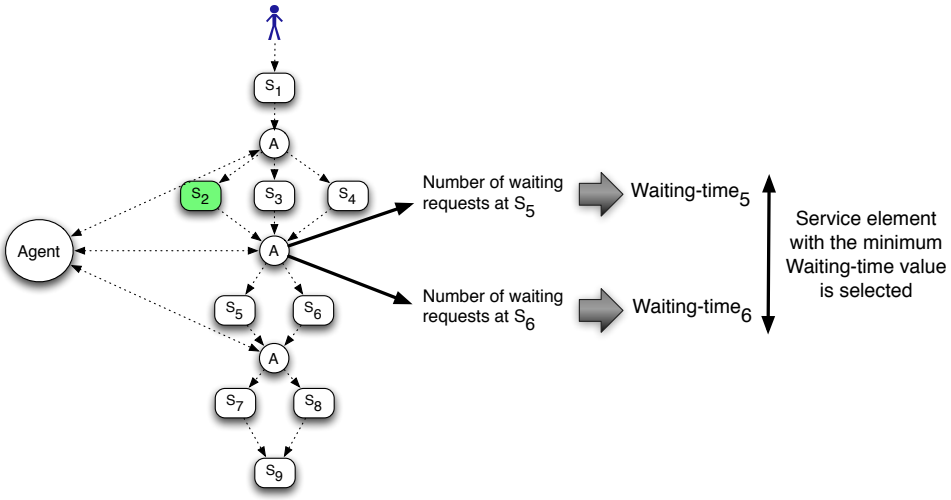


Figure 4.8: The service domain representation along with satellite agents in a dynamic setting

$$Service\ element\ selected \Rightarrow \textit{Service element with}\ min(\frac{\textit{number of requests in queue}}{\textit{completion-rate}}) \tag{4.5}$$

A simple example demonstrating service element selection in a dynamic scenario on the basis of waiting-time is shown in Figure 4.9. The service element selected in the example is the one which has the smallest ratio of request queue-length to completion-rate.

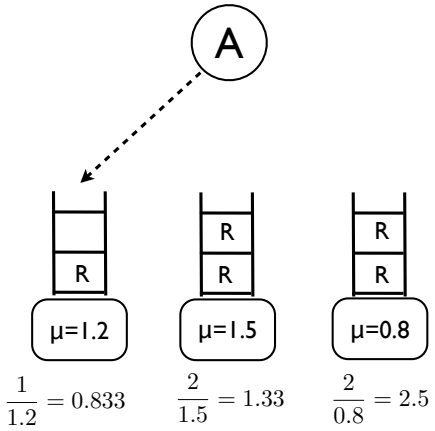


Figure 4.9: An example demonstrating the selection decision in a dynamic scenario

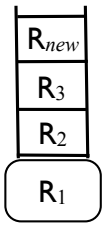
4.3.1 Mathematical verification

The technique proposed for selecting service elements, such that the total waiting-time for the composite application is minimized, is verified through mathematical

induction [75]. We first show that the technique enables the selection of the service element with the smallest waiting-time when the number of functionality levels is just 1. This is the *basis* of the induction. In the *inductive* step, we show that if the technique works correctly for n functionality levels, it will also work correctly for $(n + 1)$ functionality levels.

Basis of the induction

Assume there is a service application with only one level of functionality ($n = 1$). Each service element at the functionality level has a certain completion-rate value and a certain number of service requests waiting in a queue. According to the proposed technique, the best service element to choose on the basis of the waiting-time is the one with the smallest ratio of request-queue length to completion-rate.



$$Waiting\text{-}time_{R_{new}} = Completion\text{-}time_{R_1} + Completion\text{-}time_{R_2} + Completion\text{-}time_{R_3}$$

Figure 4.10: Waiting-time for a service request

Assume a new request arrives and joins the queue of a service element S that already has m number of requests waiting in the queue, and has a completion-rate of C requests per unit time. The waiting time for this new request is the sum of the completion-time for the request being currently serviced and the completion time of all the requests waiting ahead of it (the new request) in the queue as shown in Figure 4.10 (assuming negligible transition time from the queue to the service element).

Completion-rate of service element S is C requests completed in 1 unit of time

$\Rightarrow$ *1 request completed in* $\frac{1}{C}$ *units of time*

$\Rightarrow$ *m requests completed in* $\frac{1}{C} * m$ *units of time*

From the above reasoning we see that m requests are completed by the service element in $\frac{m}{C}$ units of time. The waiting-time for the new request is therefore $\frac{m}{C}$ since the total number of requests in the queue to be serviced before it is m. The waiting-time of the request is thus minimized by minimizing the value of $\frac{m}{C}$ which is the basis of selection in the proposed selection technique.

The proposed technique therefore works correctly in finding the service element with the minimum waiting-time when the number of functionality levels in the application is 1.

Inductive step

Assume now that the proposed technique is successful in finding the set of n service elements with the minimum waiting-time when the number of functionality levels are n. Given this assumption, the inductive step involves finding the set of service elements from $(n+1)$ functionality levels with the minimum waiting-time. This is shown in Figure 4.11.

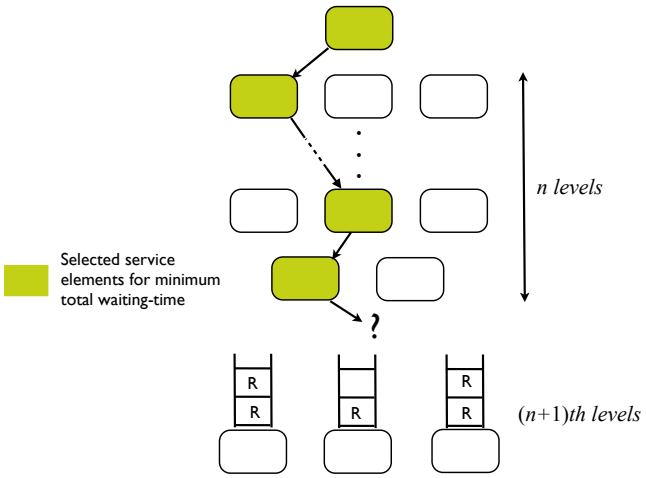


Figure 4.11: The inductive step of the mathematical induction procedure

Suppose the minimum waiting-time for n functionality levels of the domain is $WTmin_n$. According to the proposed technique, the service element selected at the $(n+1)^{th}$ level would be the one with the minimum value of the ratio $\frac{m}{C}$, where m is the request queue-length, and C is the completion rate of the service element. The total waiting-time of the composite application up to the $(n+1)th$ level is then given by equation (4.6).

$$WT_{n+1} = WTmin_n + min(\frac{m}{C}) \tag{4.6}$$

We know from the basis of the induction that the ratio min($\frac{m}{C}$) gives the service element with the smallest waiting-time at the $(n+1)^{th}$ level. $WTmin_n$ is the assumed smallest waiting-time value up to the n^{th} level. The sum of the two must therefore be the smallest waiting-time for the application up to the $(n+1)^{th}$ functionality level.

4.4 Summary

This chapter presents two techniques for calculating the selection of service elements with the aim of minimizing the waiting-time of the requests. The first technique is suitable for a static scenario where the service elements are selected before-hand, and the second technique is when the service elements are selected dynamically when the service request arrives at the given functionality level. The static scenario represents a situation where service providers put together an efficient service application ahead of time and offer it to customers having similar preferences. The technique utilizes concepts from queueing theory. Validation of the static technique is done by running simulations on the service domain and calculating the average waiting-time at the service elements by observation. The ranking of service elements in terms of average waiting-time at each level on the basis of the simulation results are compared with those on the basis of the proposed technique. The ranking results are found to be similar in most cases.

The waiting-time in the dynamic technique is calculated on the basis of data returned by 'satellite' agents that are present at each functionality level. The satellite agents keep track of the number of requests waiting at each service element. The service element is selected at run-time when the request arrives at the functionality level on the basis of the current number of requests waiting at each service element and the completion-rate of the service elements. The service element chosen is the one with the minimum ratio of request queue-length to completion-rate. The technique is verified mathematically by induction.

Chapter 5

Reputation

In this chapter we examine the reputation attribute and present a technique to assess its value in the context of dynamic service composition.

Reputation is a non-functional attribute that is susceptible to varied interpretations. In this thesis we choose to interpret the reputation of a service as a function of feedback from customers who have used the service. The risk in taking such an approach is that the factors that influence customer feedback are seldom consistent. This is more the case when the feedback becomes the basis for comparing different services. Reputation assessed using feedback is therefore often not considered accurate. To overcome this we develop a novel technique that works towards compensating for the variations in factors that influence customer feedback. We utilize multiple regression [68] to level out the differences in the influencing factors while computing the service reputation using customer feedback. The technique is validated using data on three prominent hotel chains and by comparing our results with those of an acclaimed hotel ranking survey.

5.1 Related Work

This section is a discussion on various existing techniques of reputation assessment. The target entities whose reputations are assessed are not necessarily services. Most of the techniques discussed are, however, quite generic and can easily be modified to be used for the reputation assessment of services.

Malik *et al.* [80] introduce the RATEWeb reputation assessment system for web-services. In this system, for every Quality of Service (QoS) parameter there is a provider promised value $QRef_p$, a customer expected value $QRef_r$, and the actual delivered value of the QoS parameter $QRef_d$ as perceived by the invoking customer. When a customer invokes a service, the difference between the perceived delivered QoS value $QRef_d$ and the promised value $QRef_p$ determines the reputation of the service as perceived by the respective customer. The overall reputation of the service is computed as an aggregate of the perceived reputations of all the customers that invoke the service. Bucheggar *et al.* [81] have a similar approach to

reputation assessment; however they describe the setting of their system in a peer-to-peer rather than a customer-provider scenario. Each peer, based on the perceived performance of another, provides a reputation value to the latter.

Fagini *et al.* [82] explore an interesting method to quantify qualitative expressions for reputation. The purpose of doing this is to incorporate reputation into an expression for calculating the rating of a bank. In the method proposed, the reputation of the bank is given a value depending on the kinds of words used to describe the bank in a leading Italian newspaper. The words include adjectives, adverbs, prepositions etc. and an 'opinion mining algorithm' is utilized to assign the value. Morinaga *et al.* [83] have a similar approach in which they rate the reputation of products available on various web pages. The opinion of customers on the target products are collected and the opinions are classified as positive or negative based on certain linguistic rules. Depending on the number of positive or negative results that a product receives, it is assigned a quantitative value that reflects its reputation.

Chang *et al.* [84] present a generic description of reputation through the use of ontologies. They present two types of reputation ontologies. The 'basic ontology' provides a "simplistic view" of reputation whereas the 'advanced ontology' comprises the use of statistical techniques to more comprehensively describe reputation. The target domain of these ontologies is mainly web-based systems, but the description may be used for other kinds of systems as well.

An interesting technique for assessing the reputation of an entity in a social network setting is proposed by Pujol *et al.* [85]. The proposed method does away with the practice of using the feedback of interacting entities to assess the respective reputation values. Instead, the 'location' of an entity in the social network gives an idea of its reputation. An entity that is heavily connected is one that is popular and has a larger number of entities wanting to interact with it. Such an entity is deemed to have a better reputation than one that has lesser number of connections in the social network.

Each of the techniques discussed assigns a reputation value to a service without taking into account factors such as price, service category, and other factors that can influence the feedback. This is an important restriction especially when services are compared with each other on the basis of reputation. It is akin to, for example, comparing the reputations of two airlines by using the feedback of the 'executive class' passengers of one airline and the 'economy class' passengers of the other. This is the kind of restriction that we attempt to overcome in our work. Our technique takes into account all such factors that can influence the feedback of the customer and subsequently assesses the reputation from the collected feedback.

5.2 Reputation Assessment

The assessment of reputation of service elements in the thesis is developed on the basis of feedback collected from customers that use the service element. The feedback can be in quantitative terms wherein the customer rates the service element out

Table 5.1: Hypothetical feedback values for the airline example.

Travel class	Route	Feedback (out of 6)
Economy (1)	Vancouver-Beijing (1)	2
Economy (1)	NYC-London (2)	3
Economy (1)	Toronto-Frankfurt (3)	4
Economy (1)	Paris-Delhi (4)	5
Business (2)	Vancouver-Beijing (1)	3
Business (2)	NYC-London (2)	2
Business (2)	Toronto-Frankfurt (3)	1
Business (2)	Paris-Delhi (4)	0
Premium (3)	Vancouver-Beijing (1)	6
Premium (3)	NYC-London (2)	3
Premium (3)	Toronto-Frankfurt (3)	0
Premium (3)	Paris-Delhi (4)	4

of a certain number of maximum points (example out of 5 or 6). Alternatively, the customer can express his/her feedback in words which can be appropriately converted to a value out of 5 or 6. The higher the feedback value of a service element, the better its reputation.

When service elements need to be *compared* on the basis of their respective reputation values, a little more care needs to be taken. To ensure that the service elements are compared in a fair way, the customer feedback is considered keeping all the influencing factors at equivalent levels. Continuing with our example, when two airline services are compared, as noted earlier it would be unfair to compare feedback from the economy class customers of one airline and the executive class customers of the other. Similarly the time of year when the feedback is collected from the customers of either airlines needs to be the same. Also the route on which the customers are traveling needs to be similar. Each of these factors and more influences the overall customer experience and thus the subsequent feedback. Even the profile of the customer giving the feedback needs to be taken into account while drawing comparisons between services based on reputation.

To incorporate all the factors in coming up with reputation values, we utilize multiple linear regression [68]. Multiple linear regression is useful in a scenario where a dependent variable is a function of a number of independent variables. The regression is utilized to 'fit' a line, plane, or hyper-plane (depending on the number of independent variables) into the plot of the dependent variable against the independent variables. The line, plane, or hyper-plane fitted is the one that minimizes the sum of the square of the distances from the plane to each of the plotted points.

To demonstrate the technique with a simple airline service example, assume

feedback is collected from the customers of competing airlines. While this is done, the travel class (economy, business, premium) of the customers is noted; and the travel route is also noted (we incorporate only two factors here because it is difficult to visually demonstrate a plot with more than 3 dimensions). Hypothetical values for one of the competing airlines are shown in Table 5.1. The values from the table are then plotted in the graph shown in Figure 5.1.[1] Each competing airline service has its own set of plots.

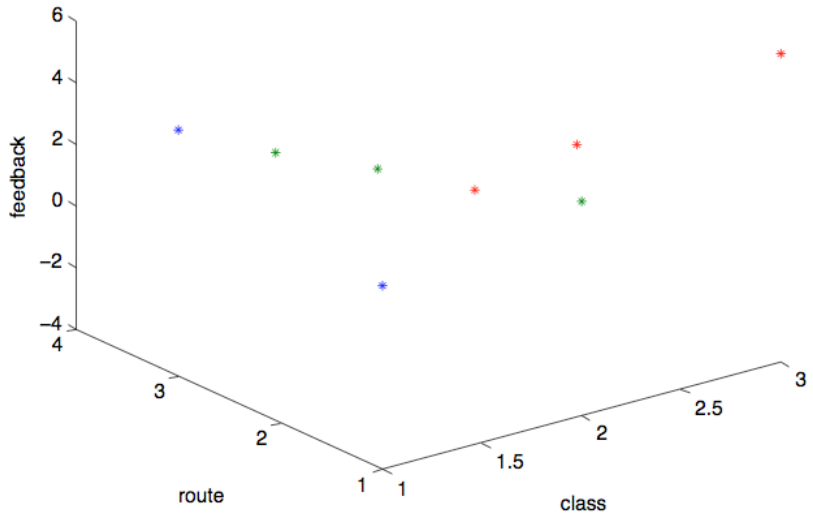


Figure 5.1: Feedback plotted the values of the influencing factors

Once the values are plotted for each of the competing services, a tool capable of performing multiple linear regression is utilized to fit a hyper-plane on the plotted points. This is shown in Figure 5.2. As mentioned earlier, the hyper-plane fitted is the one that minimizes the sum of the square of the distance of the plane from each of the points. The form of the equation representing the hyper-plane is shown in equation (5.1).

$$y = a_1 * x_1 + a_2 * x_2 + \ldots + a_n * x_n \tag{5.1}$$

The y in equation (5.1) denotes the feedback value and each of x_1, x_2, ..., x_n denote the values of the independent variables (travel class, and route in this example).

With the hyper-plane corresponding to each of the competing services in place, a reputation measure for each of the services is required such that the services can be compared. We propose that the hyper-volume under each of the fitted hyper-planes gives an idea of the reputation of the corresponding competing services. The larger the hyper-volume, the better the reputation. This makes sense because in this work

[1] the purpose of Figure 5.1 is to demonstrate the technique and therefore some of the points may not be accurately plotted

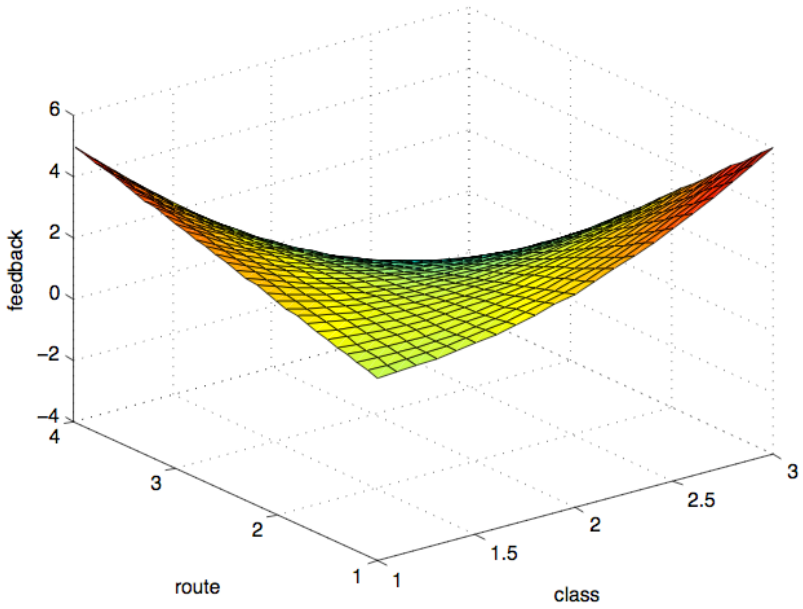


Figure 5.2: The curve 'fitted' into the feedback points by regression

we have chosen customer feedback as the only factor determining the reputation. The fact that all factors influencing the feedback are incorporated in coming up with the hyper-plane removes bias in the reputation assessment.

The hyper-volume under the hyper-plane for each of the competing services is easily calculated by integrating the corresponding equation with respect to all the independent variables. This is shown in equation (5.2).

$$V = \iint \ldots \int (a_1 * x_1 + a_2 * x_2 + \ldots + a_n * x_n) dx_1 dx_2 \ldots dx_n \tag{5.2}$$

Comparing the competing services on the basis of reputation is now just a matter of comparing this hyper-volume. The service with the largest hyper-volume is deemed to have the best reputation.

5.3 Validation

To validate the proposed technique, we utilized the simple case of the hotel service. Three 'upscale' hotel chains were selected. The customer feedback on hotels from each chain was gathered from around the world, and using this feedback the reputation of the chain was assessed using the proposed technique. The three hotels were rated on the basis of the assessed reputation and these ratings were compared with the ratings of an acclaimed survey. The conformance between our ratings and that of the survey serves to validate our technique.

Figure 5.3: Example of feedback collected from tripadvisor.com

Three upscale hotel chains were selected as representative hotel chains: 1) Sofitel, 2) Hilton, and 3) Sheraton. Feedback on hotels belonging to these chains were collected from six different cities around the world. The cities were: 1) London, 2) Paris, 3) Beijing, 4) Buenos Aires, 5) San Francisco, and 6) Cairo. Feedback was collected from the 'customer-review' section of the trip planning portal *tripadvisor* [86]. Customers express their feedback on tripadvisor in the form of a rating out of 5 besides expressing their opinion in words (an example is shown in Figure 5.3). As much feedback as available was collected for each of the hotels belonging to the three chains over the six cities mentioned. The feedback was collected for alternate months from January 2009 to November 2009. The price per room for each of the hotels at the respective time of the year was also noted. As far as possible, the prices of *equivalent* categories of rooms in the various hotels were incorporated.

The next step was to 'fit' an appropriate hyper-plane into the collected data for each of the hotel chains using regression. The independent variables in the regression were: 1) location (city) of the hotel; 2) time (month) of collecting the feedback; 3) price per room of the hotel. The dependent variable was the collected feedback, expressed as a rating out of 5. For each month, the average of the ratings collected from different customers was considered. We concede that there are many other factors that influence customer feedback besides the ones considered here, but for simplicity and for the purpose of demonstration we are only considering these three factors. Table 5.2 shows the values of the independent variables and the feedback ratings for the Sofitel group of hotels. The values of 1 to 6 for the cities respectively represent: London, Paris, Beijing, Buenos Aires, San Francisco, and Cairo. The values of 1 to 6 for the alternate months respectively represent: January, March, May, July, September, and November (all in 2009).

We utilized the regression tool available in Excel 2004 [87] to fit the hyper-plane into the collected data for each of the hotel chains. This gave us the equation of the hyper-plane fitted for the data of each of the chains. The results of regression for the Sofitel, Hilton, and Sheraton hotel chains are respectively shown in Table 5.3. The values in the column are the coefficients corresponding to location (city,

Table 5.2: Values of independent and dependent variables for the Sofitel chain.

Sofitel			
City	Month	Price	Feedback
1	6	395	4.6
1	5	395	4.6
1	4	395	5
1	3	395	4.1
1	2	395	4.8
1	1	395	4.8
2	6	320	4.4
2	5	320	4.5
2	4	320	5
2	3	320	4.6
2	2	320	4.6
2	1	320	4.5
3	6	158	5
3	5	158	5
3	4	158	4
3	3	158	4.8
3	2	158	5
3	1	158	5
4	6	236	3
4	5	236	5
4	4	236	4.6
4	3	236	5
4	2	236	5
4	1	236	5
5	6	133	2
5	5	133	3.3
5	4	133	3.7
5	3	133	3.5
5	2	133	3.3
5	1	133	3.5
6	6	206	4.1
6	5	206	5
6	4	206	4.5
6	3	206	5
6	2	206	4.3
6	1	206	4

Table 5.3: Regression results for the three hotel chains.

Sofitel	Coefficients	Hilton	Coefficients	Sheraton	Coefficients
Intercept	4.4705	Intercept	6.6159	Intercept	3.0491
X-Variable 1	-0.0704	X-Variable 1	-0.3657	X-Variable 1	0.01326
X-Variable 2	-0.0833	X-Variable 2	-0.0429	X-Variable 2	0.0498
X-Variable 3	0.0019	X-Variable 3	-0.00361	X-Variable 3	0.00301

Table 5.4: Hyper-volume values for the three hotel chains.

Sofitel	27,613.65
Hilton	43,939.33
Sheraton	23,753.85

c), time (month, m), and price (p) respectively. Equation (5.3) shows the equation of the hyper-plane for the Sofitel chain.

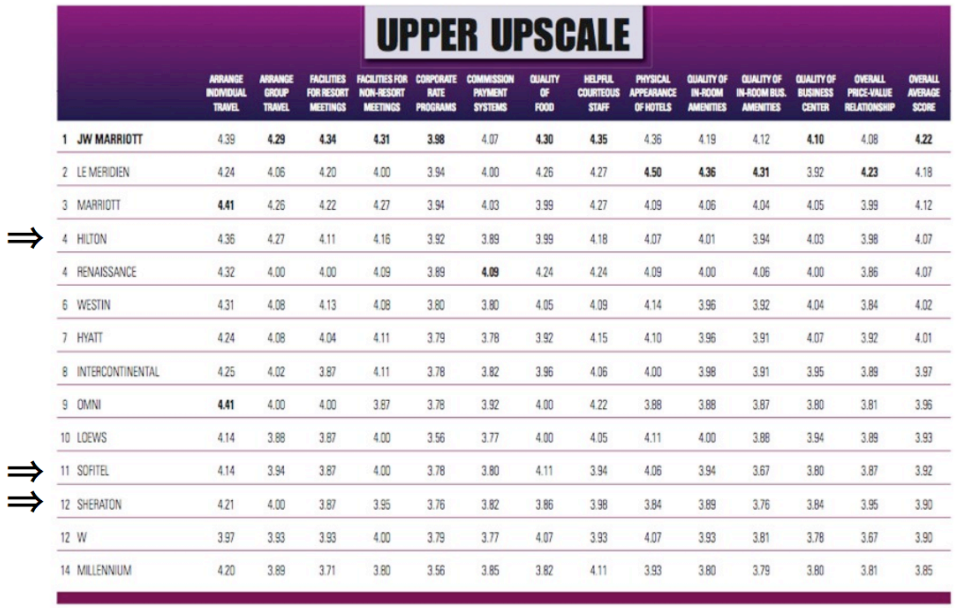
$$f = -0.0704 * c - 0.0833 * m + 0.0019 * p + 4.4705 \tag{5.3}$$

Finally, with the equation of the fitted hyper-plane for each of the hotel chains available, the hyper-volume under the hyper-plane was calculated using the expression in equation (5.2). Equation (5.4) shows the expression to calculate the hyper-volume for the Sofitel chain.

$$V = \int_{133}^{395} \int_{1}^{6} \int_{1}^{6} \{-0.0704c - 0.0833m + 0.0019p + 4.4705\} d(c)d(m)d(p) \tag{5.4}$$

The values of the hyper-volume calculated for each of the hotel chains are shown in Table 5.4. As per the proposed technique, the hotel chain that has the largest hyper-volume under its fitted hyper-plane is the one with the best reputation. The results show that the Hilton hotel chain is by far the most reputed, and the other two hotels are somewhat similar in reputation with Sofitel having a marginally better reputation than Sheraton.

We next compare our results with the *2009 U.S. Hotel Chain Survey* [88] and find an a strong conformance. As shown in Figure 5.4, the Hilton hotel chain is placed much above the other two at fourth position in this survey. Sofitel and Sheraton are much lower placed at the eleventh and twelfth positions respectively.



UPPER UPSCALE

		ARRANGE INDIVIDUAL TRAVEL	ARRANGE GROUP TRAVEL	FACILITIES FOR RESORT MEETINGS	FACILITIES FOR NON-RESORT MEETINGS	CORPORATE RATE PROGRAMS	COMMISSION PAYMENT SYSTEMS	QUALITY OF FOOD	HELPFUL COURTEOUS STAFF	PHYSICAL APPEARANCE OF HOTELS	QUALITY OF IN-ROOM AMENITIES	QUALITY OF IN-ROOM BUS. AMENITIES	QUALITY OF BUSINESS CENTER	OVERALL PRICE-VALUE RELATIONSHIP	OVERALL AVERAGE SCORE
1	JW MARRIOTT	4.39	**4.29**	**4.34**	**4.31**	**3.98**	4.07	**4.30**	**4.35**	4.36	4.19	4.12	**4.10**	4.08	**4.22**
2	LE MERIDIEN	4.24	4.06	4.20	4.00	3.94	4.00	4.26	4.27	**4.50**	**4.36**	**4.31**	3.92	**4.23**	4.18
3	MARRIOTT	**4.41**	4.26	4.22	4.27	3.94	4.03	3.99	4.27	4.09	4.06	4.04	4.05	3.99	4.12
⇒ 4	HILTON	4.36	4.27	4.11	4.16	3.92	3.89	3.99	4.18	4.07	4.01	3.94	4.03	3.98	4.07
4	RENAISSANCE	4.32	4.00	4.00	4.09	3.89	**4.09**	4.24	4.24	4.09	4.00	4.06	4.00	3.86	4.07
6	WESTIN	4.31	4.08	4.13	4.08	3.80	3.80	4.05	4.09	4.14	3.96	3.92	4.04	3.84	4.02
7	HYATT	4.24	4.08	4.04	4.11	3.79	3.78	3.92	4.15	4.10	3.96	3.91	4.07	3.92	4.01
8	INTERCONTINENTAL	4.25	4.02	3.87	4.11	3.78	3.82	3.96	4.06	4.00	3.98	3.91	3.95	3.89	3.97
9	OMNI	**4.41**	4.00	4.00	3.87	3.78	3.92	4.00	4.22	3.88	3.88	3.87	3.80	3.81	3.96
10	LOEWS	4.14	3.88	3.87	4.00	3.56	3.77	4.00	4.05	4.11	4.00	3.88	3.94	3.89	3.93
⇒ 11	SOFITEL	4.14	3.94	3.87	4.00	3.78	3.80	4.11	3.94	4.06	3.94	3.67	3.80	3.87	3.92
⇒ 12	SHERATON	4.21	4.00	3.87	3.95	3.76	3.82	3.86	3.98	3.84	3.89	3.76	3.84	3.95	3.90
12	W	3.97	3.93	3.93	4.00	3.79	3.77	4.07	3.93	4.07	3.93	3.81	3.78	3.67	3.90
14	MILLENNIUM	4.20	3.89	3.71	3.80	3.56	3.85	3.82	4.11	3.93	3.80	3.79	3.80	3.81	3.85

Figure 5.4: 2009 U.S. Hotel Chain Survey

5.4 Related Concerns

It is common for one attribute to play a larger role in influencing feedback than another. The technique presented in this chapter, however, makes the simplifying assumption that all the attributes contribute equally in influencing feedback. This restriction is easily overcome by pre-determining the nature of the hyper-plane and 'forcing' the regression tool to fit such a hyper-plane into the plotted data points. For example, in the hotel example of the previous section, one may feel that the attribute *city* has a larger influence on feedback than *month* or *price*. The nature of the equation can then be pre-determined as, for example, shown in equation (5.5).

$$f = X_1 * c^2 + X_2 * m + X_3 * p + X_4 \tag{5.5}$$

The parameter corresponding to city (c) has been squared in the equation. This results in a hyper-plane in which the city attribute value is weighted non-linearly (the square in this case) in comparison to the linear values of the other attributes.

Another concern that can be raised in the technique is that of faulty reputation assessment owing to more feedback coming from one parameter instance than another. For example, in the hotel service described previously much more feedback could be collected from San Francisco than Cairo. Further, the feedback from Cairo could be vastly different from that from San Francisco. Concerns on the reputation being faulty in such a case are misplaced because the basis of the regression is to minimize the sum of squares of the distance between the fitted plane and the plotted data points. As a result, a feedback element that is an outlier has a much smaller influence on the nature of the plane than other more closely clustered feedback elements. Therefore the reputation will not be affected much by one or two feedback

points being vastly different from the majority.

The work presented in this chapter is preliminary relative to other parts of the thesis. The main area that requires further work is the validation of the technique.A more comprehensive validation process comprising of a larger number of hotel chains with feedback collected from more varied sources would be ideal. At present only three hotel chains are compared based on feedback collected only from *tripadvisor* [86].

The use of linear regression for fitting lines and planes into the plotted feedback points is inappropriate especially when there are rapid fluctuations in feedback values. In this work we use linear regression for simplicity but more accurate 'local' regression technique like *LOWESS* [89] are desirable.

Finally, the feedback values collected from customers are assumed to be standardized in this work. This means for example that a feedback value of 3 out of 5 means the same for all customers. This is not true, some customers can be very tough in assigning feedback whereas others can be lenient. Techniques to standardize feedback values across customers need to be explored in future research and be incorporated into this work.

5.5 Summary

In this chapter a technique was described to assess the reputation of services based on feedback collected from customers. The issue with reputation assessment using feedback is that it is difficult to incorporate the variations in the factors influencing the customer feedback. This issue is of larger concern in a scenario like our composition environment where the computed reputation is a discriminating factor in comparing service elements. To overcome this we proposed a technique wherein the feedback values were plotted against the values of the influencing factors. A line, plane, or hyper-plane (depending on the number of influencing factors) was fitted to the plotted points. The area/volume/hyper-volume under the line/plane/hyper-plane gave an indication of the reputation value using feedback. The proposed technique was shown to be effective by using it to assess the reputations of three prominent hotel chains using customer feedback collected from *tripadvisor.com*. The reputation results were found to conform closely with those of an acclaimed survey.

Reputation, like the other attributes of reliability and waiting-time has been treated as a single factor for selecting services for the composition. Chapter 6 presents techniques to combine the reputation values with the values of other service attributes to create a multi-attribute service composition approach.

Chapter 6

Service composition incorporating multiple factors

In the previous chapters we examined three non-functional attributes in the context of using them individually as discriminating factors for selecting services for composition. Novel techniques were presented to calculate the values of each of these attributes. The challenge now is to bring them together such that the service selections simultaneously incorporate the values of all the service attributes.

The other challenge is to incorporate the preferences of the customers directly into the service composition process. Every customer is unique and may have dynamically changing preferences. We present a simple procedure to register the preferences articulated by customers and directly incorporate these into the service composition process. The procedure is straightforward and flexible in allowing customers to modify their preferences even after the service composition process is initiated.

Our approach brings the values of the individual service attributes and the preferences articulated by the customers together using one unifying factor called *affinity*. The selections made based on the affinity factor have the combined 'flavour' of all the service attributes and the preferences of the customer.

The sections in this chapter are sequentially structured following the order of steps in the service composition procedure. We use the motivating trip-planner example introduced in Chapter 2 to describe the service composition process. The trip-planner service domain is shown in Figure 6.1. We assume that at this stage the service selection for the *flight service* functionality is completed (*Air Canada* is selected) as depicted in the figure. The task at hand is to select an appropriate service element from the next level of functionality: the *taxi service*. Our composition approach will be demonstrated as we make the selection from the taxi service functionality.

Figure 6.1: The trip-planner scenario: Air Canada is the selected Flight service; an appropriate Taxi service element is to be selected

6.1 Step 1: Assessment of individual service attributes

This first step in our service composition approach is the assessment of the individual service attributes of the available service elements. The service attributes include the basic functionality of the service element and the non-functional attributes of the service element. Techniques for this step were discussed in detail in the previous three chapters of this thesis. Chapter 3 discussed a novel technique for the assessment of the reliability attribute in an interactive composition environment. This technique enables us to incorporate the influence of the interacting service elements effectively while computing the reliability values of individual service elements. Chapter 4 examined the waiting-time attribute of the available service elements. Two approaches: a static and a dynamic approach were presented to assess the waiting-time of the service requests at the service elements. The static approach utilizes techniques from queueing-theory whereas the dynamic approach assesses the current waiting-time at each service element simply as the ratio of the number of waiting requests to the current service completion-rate of the service element. Finally, Chapter 5 presented a novel technique for assessing the reputation of a service element based on customer feedback. The technique effectively compensates for the variations in the factors that influence customer feedback such that the reputation values computed are free of bias of any kind.

The techniques described previously and assessment techniques for the basic

functionality of the service elements are applied on each potential service element of the taxi service. This is shown for the *Taxi* taxi service element in Figure 6.2. The same assessment techniques are applied on all the other taxi service elements available.

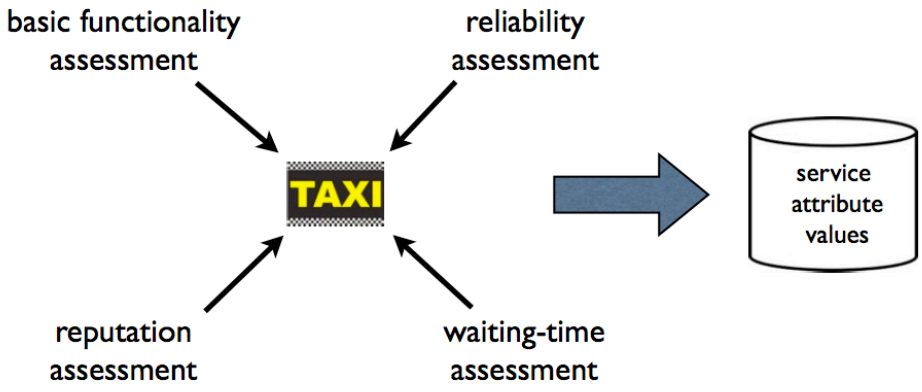


Figure 6.2: Assessment of individual service attributes

It is important to note that the techniques proposed in this thesis to assess the individual service attributes need not necessarily be used. Depending on requirements and convenience other techniques can be used for assessment of individual service attributes. Also, a different set of service attributes can be used to make the selection decisions. Techniques for calculating the values of the functional attributes of service elements are not included in this thesis because these are specific to the nature of the service and would vary depending upon the service type.

6.2 Step 2: Assessment of customer preferences

The next step in our approach is the collection of preferences of customers and incorporating them into the service composition process. This is a challenging task as customers of services are usually humans or systems controlled by humans. As such the articulation of preferences by customers becomes a complex procedure characterized by unanticipated situations. This motivated us to make the task of articulating preferences of the customer a very simple one that is easy to understand and use.

The process of collecting customer preferences is described as follows: each service attribute partaking in the service composition process is assigned a preference weight. In the thesis, we assign the following preference weight parameters to the service attributes: 1) basic functionality $\rightarrow \alpha$, 2) reliability $\rightarrow \beta$, 3) waiting-time $\rightarrow \gamma$, and 4) reputation $\rightarrow \delta$. The customer assigns values between 0 and 5 to each of the preference weight parameters. The customers are asked to view this activity as tuning a set of 'knobs' on a control panel that he/she can use to control the

degree of influence of each service attribute in the service composition. The customer is also told that the tuning parameters are relative in nature. This means that, for example: if a customer sets all the preference weights to 5, equal preference is given to all the service attributes during the process of service selection; this would result in service selections that are identical to the situation when all the attributes are assigned a preference weight of 1. The customer should therefore set high preference weights relatively on the more preferred attributes and small weights to the less preferred ones.

Figure 6.3 shows the assignment of values to the preference weights. A customer who is clear about his/her preferences directly assigns values to the preference weight parameters. A customer who is not very clear about what exactly he/she wants and is unable to set the values of the preference weights directly is assisted in doing so by the Hypothetical Equivalents and Inequivalents Method (HEIM) [42], an Operations Research technique customized by us to suit service selections. The customized HEIM technique [103] is described by means of an example in the following section.

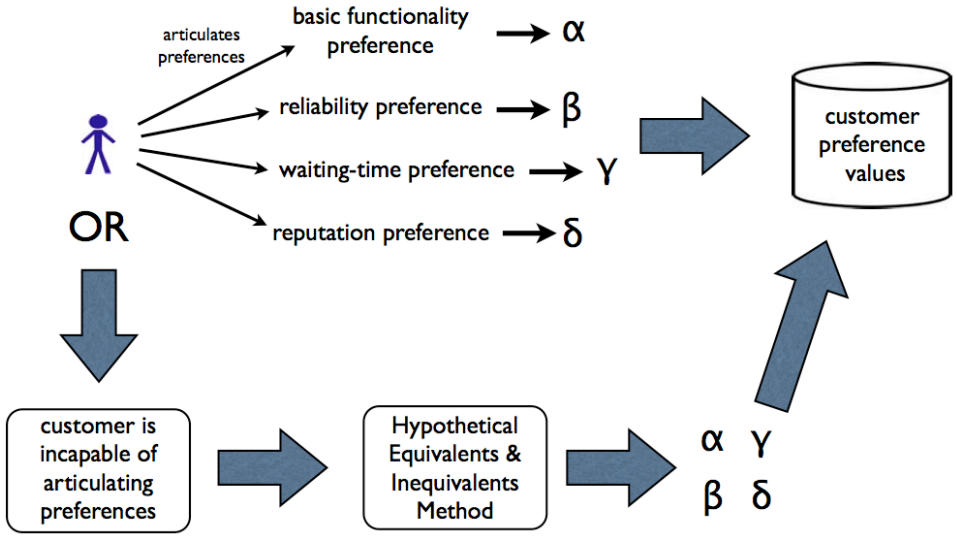


Figure 6.3: Articulation of customer preference weight values

6.2.1 Hypothetical equivalents and inequivalents method

The Hypothetical Equivalents and Inequivalents Method (HEIM) is a simple procedure that assists customers in articulating their preference weights with regard to the service attributes. The procedure comprises putting together a set of hypothetical services with random values of service attributes. We demonstrate the HEIM techniques using a simple example. Table 6.1 shows a set of 10 hypothetical services called HS_1, HS_2, ..., HS_{10} with random values for each service attribute.

Table 6.1: Hypothetical services with randomly generated attribute values

Service	Functional attribute	Reliability	Waiting-time	Reputation
HS_1	15	68%	12 min	0.15
HS_2	65	75%	69 min	0.66
HS_3	29	87%	85 min	0.14
HS_4	43	70%	54 min	0.59
HS_5	59	33%	17 min	0.88
HS_6	60	43%	78 min	0.62
HS_7	10	93%	79 min	0.42
HS_8	69	69%	50 min	0.81
HS_9	91	46%	170 min	0.73
HS_{10}	89	97%	47 min	0.7

The customer is shown this set of hypothetical services and is asked to rate relatively as many of these services as possible according to the schema: x is better than ($>$) y; x is as good as ($=$) y; x is worse than ($<$) y. Assume the customer says: 1) $HS_5 > HS_2$, 2) $HS_9 > HS_6$, and 3) $HS_4 = HS_1$. The customer ratings are stored in a repository.

Next, the random attribute values in Table 6.1 are normalized between 0 and 1 such that the best attribute value is equal to 1. This is simply done by dividing each service attribute value by the best value. For waiting-time, since the lowest value is the best, each value is divided by the best value and the reciprocal of the result is calculated. The normalized service attribute values are shown in Table 6.2. Each service attribute is assigned a parameter (w_i) which represents the preference weight of the attribute. The second row in Table 6.2 shows these parameters. A weight is assigned to each hypothetical service as the product of the normalized service attribute values and the preference weight parameter. These products are summed across all service attributes. The service weights are shown in the last column of Table 6.2.

The ratings articulated by the customer based on his/her preferences of the hypothetical services are used to form equations and inequalities in terms of the service weights in the last column of Table 6.2. This is shown in equations (6.1).

$$
\begin{aligned}
HS_5 > HS_2 &\Rightarrow 0.65w_1 + 0.35w_2 + 0.71w_3 + w_4 > 0.71w_1 + 0.8w_2 + 0.17w_3 + 0.75w_4 \\
HS_9 > HS_6 &\Rightarrow w_1 + 0.49w_2 + 0.07w_3 + 0.83w_4 > 0.66w_1 + 0.46w_2 + 0.15w_3 + 0.7w_4 \\
HS_4 = HS_1 &\Rightarrow 0.47w_1 + 0.74w_2 + 0.22w_3 + 0.67w_4 = 0.16w_1 + 0.72w_2 + w_3 + 0.17w_4
\end{aligned}
\tag{6.1}
$$

The inequalities and equation in (6.1) serve as constraints to the objective function shown in equation (6.2) that needs to be optimized. Optimizing the objective

Table 6.2: Normalized attribute values of hypothetical services and service weights.

Service	Func. attr.	Rel.	Wait.	Rep.	Service Wt.
	w_1	w_2	w_3	w_4	
HS_1	0.16	0.72	1	0.17	$0.16 * w_1 + 0.72 * w_2 + 1 * w_3 + 0.17 * w_4$
HS_2	0.71	0.8	0.17	0.75	$0.71 * w_1 + 0.8 * w_2 + 0.17 * w_3 + 0.75 * w_4$
HS_3	0.32	0.93	0.14	0.16	$0.32 * w_1 + 0.93 * w_2 + 0.14 * w_3 + 0.16 * w_4$
HS_4	0.47	0.74	0.22	0.67	$0.47 * w_1 + 0.74 * w_2 + 0.22 * w_3 + 0.67 * w_4$
HS_5	0.65	0.35	0.71	1	$0.65 * w_1 + 0.35 * w_2 + 0.71 * w_3 + 1 * w_4$
HS_6	0.66	0.46	0.15	0.7	$0.66 * w_1 + 0.46 * w_2 + 0.15 * w_3 + 0.7 * w_4$
HS_7	0.11	0.99	0.15	0.48	$0.11 * w_1 + 0.99 * w_2 + 0.15 * w_3 + 0.48 * w_4$
HS_8	0.76	0.73	0.24	0.92	$0.76 * w_1 + 0.73 * w_2 + 0.24 * w_3 + 0.92 * w_4$
HS_9	1	0.49	0.07	0.83	$1 * w_1 + 0.49 * w_2 + 0.07 * w_3 + 0.83 * w_4$
HS_{10}	0.98	1	0.26	0.8	$0.98 * w_1 + 1 * w_2 + 0.26 * w_3 + 0.8 * w_4$

function will assign relative values summing to 1 to the preference weight parameters based on the above constraints. An appropriate optimization technique such as Linear Programming [43] or the Generalized Reduced Gradient Method [44] can be used to solve the optimization problem.

$$Minimize(1 - \sum_{i=1}^{4} w_i)^2 \tag{6.2}$$

We solved the optimization problem though Linear Programming with the 'Solver' toolkit that comes built-in with Microsoft Excel (2004). The following preference weight values were obtained for the current example: $w_1 = 0.57$, $w_2 = 0.2$, $w_3 = 0.23$, and $w_4 = 0$. We normalize the preference weights between 0 and 5 (as this is the range considered in our service composition approach) by multiplying the obtained values by 5. We therefore get the following preference weight values: $w_1 = 2.85$, $w_2 = 1$, $w_3 = 1.15$, and $w_4 = 0$.

The HEIM technique can be used in this way to assign preference weights for customers who are not confident of doing so themselves based on their preferred hypothetical services.

6.3 Step 3: Combining the service attributes and customer preferences

In this step the service attribute values computed for the potential service elements in Step 1 and the articulated customer preferences described in Step 2 are brought together through a unifying factor called affinity. Affinity is a factor that is a function of: 1) the domain characteristics of the service composition environment, 2)

the individual service attribute values computed or gathered, and 3) the relative preferences with respect to the service attributes articulated by the customer.

The domain characteristics of the service composition environment comprise: 1) the inter-relationships among the service elements and 2) the dynamic nature of the domain characterized by the regular entry and exit of new service elements. The inter-relationships among the service elements is quantitatively expressed by a factor called coupling and the dynamism of the domain by a factor called availability. Coupling and availability are discussed in more detail in the following sub-sections.

6.3.1 Coupling

Service selection for composition is strongly influenced by the relationships between service elements. There is a strong inclination in making a service request to select a service element that has a good relationship with the currently used service element. To elucidate the notion of coupling between services, suppose in our motivating trip-planner example, the airline service that is currently in use *Air Canada* has a special business relationship with the *Colts Cabs* taxi service. This relationship could include services such as automatic transfer of checked-in luggage or personalized pick-up for customers at airports. A customer using Air Canada for flying will therefore have a stronger inclination to choose Colts Cabs over other taxi services if all other factors are considered equal.

To express such relationships quantitatively we use a factor called *coupling*. Coupling was briefly discussed in Chapter 3 while discussing the reliability attribute. A coupling value is assigned to each pair of interacting services. The value of coupling for a pair of service elements can range from 1 to ∞. A coupling value of 1 indicates no special bonding between the services and higher values indicate strong relationships. In this thesis we assign coupling values to interacting service pairs assuming these values are available in advance. We do not propose techniques to compute in real-time coupling values between interacting service elements.

6.3.2 Availability

The dynamism of the service domain characterized by frequent entry and exit of service elements poses a problem in effectively forming service compositions. This is because of the uncertainty associated with the presence of a selected service element in the domain at the time of composition. To overcome this we use a factor called *availability*. An availability parameter is associated with every service element in the domain. The parameter can take a value of 0 or 1. A value of 1 for the availability parameter for a given service element at a given point of time indicates that the service element at that point of time is available in the domain for selection. An availability value of 0 indicates the absence, irrespective of whether temporarily or permanently, of the given service element from the service domain at the given point of time.

6.3.3 The affinity factor

Affinity is the unifying factor that combines the coupling and availability factors described, the service attributes individually assessed, and the articulated customer preference weights. The affinity factor is computed by a formula that uses a combination of all these factors and it can be easily modified to include or exclude factors. An affinity value is calculated for every pair of interacting service elements in the domain. The computed affinity value from a service element i to a service element j expresses the likelihood that service element j would be selected next for the composition given that the currently selected service element is i. A large value of $affinity_{ij}$ indicates a higher likelihood of selection of j and vice-versa. The values of affinity can range from 0 to ∞. A value of 0 for $affinity_{ij}$ indicates no likelihood of the selection of j which is usually because j is unavailable.

The computation of the affinity value from service element i to j is shown in equation (6.3). The affinity factor includes values of the four service attributes described besides the coupling and availability factors. Each of the service attribute values has an exponent equal to the preference weight assigned to that attribute by the customer. In the equation 'basic func' is the short form for basic functionality, 'reliab' for reliability, and 'reput' for reputation.

$$affinity_{ij} = coupling_{ij} \cdot \frac{\{basic\ func_j\}^{\alpha} \cdot \{reliab_j\}^{\beta} \cdot \{reput_j\}^{\gamma}}{\{waiting\text{-}time_j\}^{\delta}} \cdot availability_j \tag{6.3}$$

The asset of the affinity model is its simplicity. The affinity equation facilitates the straightforward introduction of a new service attribute into the service composition process at run-time and/or the removal of a service attribute from the process. This is done by introducing a service attribute in the numerator of the affinity equation if a large value of the attribute is desirable. Conversely, the attribute is introduced in the denominator of the equation if a small value of the attribute is desirable. This can be seen in equation (6.3). The preference for an attribute, as mentioned earlier, can be incorporated by adjusting the exponent value of that service attribute. A service attribute that is important for a customer is given a high exponent value and vice-versa. A service attribute can be removed from consideration in the service composition process by simply setting its exponent value to 0.

The calculation of the affinity factor for the *Taxi* service element from *Air Canada* is exhibited in Figure 6.4. The service attribute values that were calculated for each of the available taxi services and the preference weights articulated by the customer are drawn from a repository where they were stored.

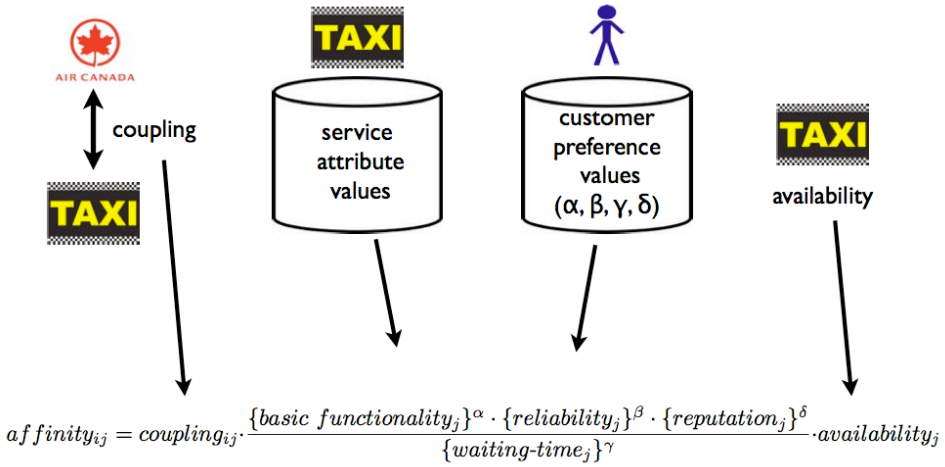


Figure 6.4: Combining all the factors in the affinity equation

6.4 Step 4: Selection of service elements for composition

The final step in the service composition approach is the selection of the appropriate service element corresponding to each functionality. The affinity factor computed in Step 3 enables the selection of the appropriate service element based on the coupling and availability factors, individual service attribute values, and customer preferences. We describe two approaches to service selection using the affinity factor. The first approach is a Greedy approach [77] which is dynamic in nature. This is the approach we recommend for a dynamic composition and is adopted by us in our experiments in the next chapter.

The second approach to selecting service elements using the affinity factor is a Dynamic Programming approach [77] and is relatively static. We have included a discussion of this approach here because it could be of interest to a reader who wants to use the Affinity Model for static compositions effectively. We do not use the Dynamic Programming approach in our composition model.

6.4.1 The Greedy approach

In the Greedy approach of service selection, the first step is the computation of affinity between the service element currently selected and all the service elements providing the next desired functionality. This is demonstrated in Figure 6.5 where the service currently selected is *Air-Canada* and the affinity values are calculated between Air-Canada and all the service elements with the taxi service functionality. From the taxi service elements the one that has the largest value of affinity with Air-Canada is the service element selected. In Figure 6.5 the *Yellow cab* service is shown to have the largest affinity value and is selected. This greedy approach is

continued through the subsequent functionalities (e.g. hotel service selection) and the service composition is dynamically formed.

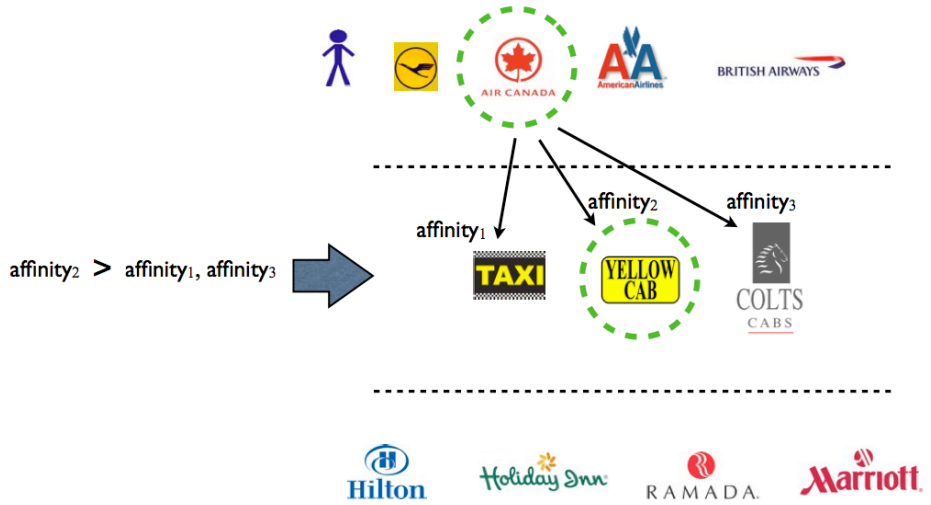


Figure 6.5: Service selection based on the Greedy approach

The Greedy approach to service selection is very simple and is based on successfully finding the local optimal solution. There is always a possibility that the composition formed through the Greedy approach is far from the global optimal if the future available service requests were known. Unfortunately they are not known and with the dynamically changing service domain, the changing service attribute values, and the flexibility afforded to customers to seamlessly modify their preferences the Greedy method of selecting provides a promising solution.

6.4.2 The dynamic programming approach

In a static service composition environment, the Dynamic Programming approach to service selection is more appropriate. The Dynamic Programming approach seeks to make service selections that are near the global optimal in terms of the computed affinity values. This means that unlike the Greedy method which locally selects the service with the largest affinity value, the Dynamic Programming approach makes service selections such that the sum of the affinity values between all the selected services of the composition is maximized. This is only possible in a static environment where all the service selections are done before the customer begins to use the composition.

The Dynamic Programming approach involves calculating, in advance, the affinity values between every pair of services that can possibly interact. The next step is to go through the domain one functionality at a time and store the maximum summation of affinity values. It is easier to explain the procedure through a simple example. Assuming we have a simplified trip-planner example domain shown in

Figure 6.6 with hypothetical affinity values calculated and labeled on the connecting arrows (a smaller domain is shown for clarity and ease of understanding). We use a matrix called *affinity* to store the maximum summation of affinity values and another matrix called *trace* to store the path that is followed to get the maximum summation of affinity values. We trace the domain in Figure 6.6 in the following manner:

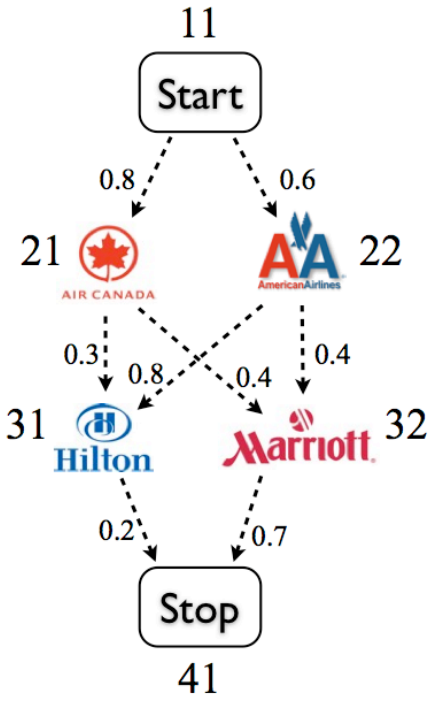


Figure 6.6: Simplified Trip-planner example for demonstrating the Dynamic Programming composition approach

The maximum affinity to the *Start* state labeled as 11 is 0 so we add this to the *affinity* matrix and since this the first state with nothing before it we store 'null' as the first value of the *trace* matrix.

$$affinity = \begin{bmatrix} 0 & \dots \\ \vdots & \ddots \end{bmatrix} trace = \begin{bmatrix} null & \dots \\ \vdots & \ddots \end{bmatrix}$$

The next service is 21. The affinity sum to this service is (from Figure 6.6): $0 + 0.8 = 0.8$, and the immediately preceding state is: 11. These are stored in the respective matrices at element 21 (2nd row, 1st column) corresponding to the service number. Similar calculations are done for service 22 and the entries are made accordingly in the respective matrices,

$$affinity = \begin{bmatrix} 0 & \dots & \dots \\ 0.8 & 0.6 & \dots \\ \vdots & \vdots & \ddots \end{bmatrix} trace = \begin{bmatrix} null & \dots & \dots \\ 11 & 11 & \dots \\ \vdots & \vdots & \ddots \end{bmatrix}$$

At the next level, for service 31, there are two alternative paths, one from 21 and the other from 22. The affinity summation from 21 is: $0.8 + 0.3 = 1.1$ and from 22 the affinity summation is: $0.6 + 0.8 = 1.4$. The larger of the two is selected (i.e. the path from 22) and the matrices are populated accordingly. The same is done for service 32. The matrices become,

$$affinity = \begin{bmatrix} 0 & \dots & \dots \\ 0.8 & 0.6 & \dots \\ 1.4 & 1.2 & \dots \\ \vdots & \vdots & \ddots \end{bmatrix} trace = \begin{bmatrix} null & \dots & \dots \\ 11 & 11 & \dots \\ 22 & 21 & \dots \\ \vdots & \vdots & \ddots \end{bmatrix}$$

Finally, for the *Stop* state labeled 41, the alternative paths have summation values: $31 \rightarrow 1.4 + 0.2 = 1.6$, and $32 \rightarrow 1.2 + 0.7 = 1.9$. The path from service 32 is larger and is selected.

$$affinity = \begin{bmatrix} 0 & \dots & \dots \\ 0.8 & 0.6 & \dots \\ 1.4 & 1.2 & \dots \\ 1.9 & \dots & \dots \end{bmatrix} trace = \begin{bmatrix} null & \dots & \dots \\ 11 & 11 & \dots \\ 22 & 21 & \dots \\ 32 & \dots & \dots \end{bmatrix}$$

The bottom-up tracing comprises simply tracing the *trace* matrix from the bottom row upwards. The element $(4, 1)$ in the *trace* matrix corresponding to state 41 has the value 32, therefore service 32 is the selection just before 41. The next element inspected in the *trace* matrix is the one corresponding to service 32 i.e. element $(3, 2)$. This element is found to point to 21. This process is continued until the initial functionality is reached. The optimal path i.e. the one with the maximum summation of affinity values is: $11, 21, 32, 41$.

We repeat that the Dynamic Programming approach described here is suitable for static service compositions. It cannot be used in dynamic service compositions and we neither use it in our composition model nor in the experiments described in Chapter 7.

6.5 Summary

In this chapter we described the step-wise procedure followed in our approach to achieving an appropriate service composition dynamically. The composition approach is based on an affinity model which incorporates the values of the service attributes of the available service elements, the preferences of the customer, and the coupling and availability factors. The affinity value calculated is then utilized with a Greedy approach wherein service elements that have the largest affinity values with preceding service elements are selected. Service selections are done at all desired functionalities in this way and a composition is formed dynamically.

We also described the Dynamic Programming approach as an alternative to the Greedy approach in a situation where the service composition process is static. We

emphasize that the Dynamic Programming approach cannot be used in dynamic service compositions and has not been used in our model.

Chapter 7

Validation

In Chapter 6 we introduced the Affinity Model which combines the values of the service attributes and customer preferences with the coupling and availability parameters to express the likelihood of selecting service elements for the specified functionalities quantitatively. In this chapter,[1] we seek to validate the presented Affinity Model for service composition. A good validation approach would be to assess its effectiveness in comparison to the selection behavior of actual customers. To do this, an ideal scenario would comprise a domain of real service elements with the service attribute values of each service element available to us. Real customers would be asked to make their selections and put together a service composition in this domain. Simultaneously, the Affinity Model would be used to form a service composition. A composition by the Affinity Model comparable to the ones put together by real customers would validate the efficacy of the model. This validation would be to the extent that the Affinity Model can automatically replicate the intuition and decision making capabilities of a real customer by simply using a set of parameter values articulated by them.

In reality though, especially in an academic environment, it is difficult to have access to a domain of service elements that could be used to conduct our experiments. Even if we should have access to such a domain, access to data that would allow us to compute the service attribute values is even more difficult. Furthermore, legal and governance issues in an inter-organizational service environment would make it impracticable to conduct experiments.

To overcome these limitations, we simulated a service composition environment in the form of a game called the *Ambitious-Traveler*. The Ambitious-Traveler game comprises a domain of transportation services. These transportation services are functionally diverse in terms of the cities between which they can transport the requesting customer. Each pair of cities has a set of transportation services from which selections are made. The selection of a transportation service is on the basis of the values of its service attributes. The aim while playing the Ambitious-Traveler game is to make judicious transportation selections in order to satisfy a certain

[1] A preliminary version of this chapter has been published in the Proceedings of the International Conference on Knowledge Management and Information Sharing, 2010 [101].

traveling goal.

The Ambitious-Traveler game was envisaged by us and it was developed by us in collaboration with Zak Turchansky, a summer student in the Department of Computing Science, University of Alberta.

We conducted experiments with the Ambitious-Traveler game that involved a set of human volunteers who were asked to play the game making intuitive transportation service selections on the basis of the service attribute values. The same game was simultaneously played by the Affinity Model. The scores (which comprised of the distance traveled and conformance to articulated preferences) produced by the Affinity Model were compared with those of the human volunteers. Comparable scores by the model serve to validate its efficacy as will be described in more detail in the chapter.

Section 7.1 discusses the Ambitious-Traveler game in more detail. Section 7.2 discusses our experiments using the game and analyze the results obtained.

7.1 The Ambitious-Traveler game

The Ambitious-Traveler is a single-player game in which a player is driven by a particular traveling goal. The main goal used in our simulation is for a player to travel as far as possible given a limited amount of time and money. There are a set of cities in Europe, shown in Figure 7.1, between which players can travel. To travel between these cities, the player has to choose from a set of transportation services. There are four transportation services available between every pair of cities: 1) a flight service, 2) a car service, 3) a train service, and 4) a boat service. Each of the four services performs the same basic functionality: to transport the player between the respective pair of cities. The four transportation services can differ in one or more of the following service attributes: 1) reliability, 2) waiting time, 3) reputation, and 4) cost of the service. The first three attributes are non-functional attributes discussed in detail in Chapters 3, 4, and 5 of the thesis. The remaining attribute, cost, is considered non-functional as described earlier in Section 2.1 because it can be associated with any service. However, it also can be considered functional in nature because clients often perceive the cost of a service as directly related to the value of the functionality provided by the service.

7.1.1 Factors influencing the selection decisions

The player playing the Ambitious-Traveler game has to travel a route through the cities using the available transportation services. The primary goal for our experiments is to travel as far as possible with a limited amount of time and money. An important constraint to the game is that a player can only visit a city once. During the play of the game, two decisions are taken at each city by the player: 1) which city to travel to next, and 2) which transportation service to use. These decisions are based on the distance between the cities (a static functional attribute) and the

Figure 7.1: European cities used in the Ambitious-Traveler game

four dynamic service attributes of the transportation service mentioned earlier: 1) reliability, 2) waiting time, 3) reputation, and 4) cost. Each of these attributes has an associated trade-off that the player needs to consider before making a transportation service selection. These are discussed in the following sub-sections.

Distance

Distance is expressed in kilometers in the game. It is a static attribute between a pair of cities and does not change at any point of the game. It is assumed that the distance values are available in advance.

Distance has an important role to play in the decision the player takes on which city to travel to next. Ideally the player would like to travel to a city that has the largest distance from the current city, however owing to other factors distance sometimes has to take a backseat in the decision making process.

It is noteworthy that distance only plays a part in determining the city to travel to next. It does not influence the decision on the type of transportation service (flight, car, train, or boat) used for traveling. This is because once the destination city is determined, all the available transportation services cover the same distance.

Reliability

Reliability is another attribute considered by the player while selecting the city to travel to and transportation service to get there. Reliability of a service is a non-functional attribute that expresses the likelihood of the service successfully performing its allocated function without failing. Reliability of a transportation service in the Ambitious-Traveler game is expressed in terms of its fail-rate. Fail-rate is the number of times the service fails on average in unit time. Each transportation service starts with a random fail-rate value which is revised every time the service fails. To keep the fail-rate of a service from spiraling to a very large value, it is brought back to its original value when it exceeds a fixed threshold.

It is in the interest of a player of the Ambitious-Traveler game to select a service of high reliability because it increases the chances of the player of reaching the destination city without the transportation service incurring a failure. A failure in the transportation service during the player's transit has the following implications: the player loses the money spent on the transportation service, the player loses the time spent in transit, and the player remains at the city of origin. When selecting a service that is highly reliable one must consider that the service could be very expensive and possibly slow (long travel time). As with all the service attributes the customer has to take a balanced approach when selecting which transportation service to use.

To create a simulated environment characterized by services that fail occasionally, we make the transportation services fail randomly at rates equal to their determined fail-rates. The failure is instantaneous and immediately after the failure the transportation service is back to performing its task. There are no implications of a failure if the failing service is not transporting a player. On the other hand, if a player is in transit on a service when the service fails the player loses the time and money invested on the service and remains at the city of origin. The player cannot then use the same transportation service to go to the same destination from the current city again.

Waiting time

The next non-functional attribute we discuss is waiting-time. Waiting-time is the time that a player has to wait at a transportation service before getting served. Each transportation service in the Ambitious-Traveler game can only transport one player at a time, hence it is not uncommon in the game for queues of waiting players to be formed. The waiting time for a player in the game is dynamically calculated as the product of the number of players ahead of the player in the queue and the transit time of the service from the current city to the destination and back. This is because the transportation service can transport only one player at a time and the next player has to wait for the transport mode to return before starting the trip.

A player prefers to choose a transportation service that has a short queue or no queue at all because it reduces or avoids waiting time for a transportation service.

This is important because the time allocated to a player is limited and should be used judiciously. On the other hand, services with shorter queues could be more expensive. The customer has to, once again, take all the trade-offs into account before taking a decision.

To simulate a realistic queueing scenario in the game we have a large number (1,000) of synthetic players playing the game simultaneously with the actual player. These synthetic players are generated randomly and have randomly assigned preferences. The synthetic players, like the actual players, travel between cities automatically selecting transportation services. The large number of synthetic players results in queues forming to get access to transportation services. It is common therefore for the actual player to wait in queues to avail himself/herself of a transportation service.

It is noteworthy that the waiting time attribute subsumes another important attribute of the transportation services: travel time. The waiting time of a player in a queue includes the travel time of those ahead of it in the queue. This is why travel time has not been explicitly considered as another attribute influencing the service selection decisions of the players.

Reputation

Ideally reputation of a service is a function of feedback from players that have previously used that service. A high reputation service is one that players prefer for reasons that can often not be expressed or quantified. Typically high reputation can be associated with a good service experience.

In the Ambitious-Traveler game we do not use player feedback to compute reputation because: 1) the number of players that have played the game is small and therefore the number of feedback elements would not be enough for accurate reputation assessment, and 2) the game is fast paced and players do not have the luxury to actually experience the services and provide accurate feedback.

To calculate reputation we simply express reputation as a function of the other attributes of the service. A good set of attribute values overall is expected to provide a good service experience to the players. Transportation selections made on the basis of reputation therefore do not have an immediate and obvious effect on the status of the game. More implicitly however, choosing high reputation services does positively affect the outcome of the game and vice-versa.

Cost

In the game, cost values are assigned to services based on the nature of the service and the distance of the cities. Cost values are typically dynamic and are dependent upon various dynamic factors. In the current version of the game, for simplicity we treat cost as a static attribute . We collected cost information in Canadian dollars from sources such as Expedia [92], and EuRail [93]. Cost values that were

not available were assigned approximate values on the basis of common sense and general experience.

Every transportation service in the Ambitious-Traveler game has a cost associated with it. The player needs to pay proper attention to cost because the money allocated for travel is limited. Shorter distances and slower services are usually associated with lower costs. If a player chooses a short distance to preserve money, the main objective of the game – to cover as much distance as possible – is compromised. If the player selects slower services, time is wasted which is also a limited resource in the game. On the other hand, larger distances and faster services are expensive and the player ends up using a large amount of money if either of these is selected. The player therefore needs to weigh carefully the options with respect to cost before selection decisions. Figure 7.2 shows a screen-shot of the Ambitious-Traveler game with service attribute values of the available transportation services.

```
Please choose where you would like to go.

1 - Paris (343km)
Plane - (0.41 Hours, Cost: $200, Reputation: 3.26/5, Waiting-time: 11.07 Hours, Fail-rate: 2.20%)
Car - (2.05 Hours, Cost: $66, Reputation: 2.34/5, Waiting-time: 10.25 Hours, Fail-rate: 1.08%)
Train - (1.23 Hours, Cost: $50, Reputation: 4.04/5, Waiting-time: 19.68 Hours, Fail-rate: 0.72%)
Boat - (3.28 Hours, Cost: $28, Reputation: 3.01/5, Waiting-time: 1 Days and 2.24 Hours, Fail-rate: 0.30%)

2 - Berlin (929km)
Plane - (1.16 Hours, Cost: $180, Reputation: 3.41/5, Waiting-time: 11.60 Hours, Fail-rate: 2.40%)
Car - (5.80 Hours, Cost: $60, Reputation: 2.42/5, Waiting-time: 11.60 Hours, Fail-rate: 1.73%)
Train - (3.48 Hours, Cost: $45, Reputation: 4.23/5, Waiting-time: 20.88 Hours, Fail-rate: 1.22%)
Boat - (9.28 Hours, Cost: $25, Reputation: 3.16/5, Waiting-time: 1 Days and 3.84 Hours, Fail-rate: 0.60%)

3 - Barcelona (1146km)
Plane - (1.42 Hours, Cost: $310, Reputation: 2.38/5, Waiting-time: 7.10 Hours, Fail-rate: 2.60%)
Car - (7.10 Hours, Cost: $103, Reputation: 1.80/5, Waiting-time: 7.10 Hours, Fail-rate: 2.38%)
Train - (4.26 Hours, Cost: $77, Reputation: 2.88/5, Waiting-time: 12.78 Hours, Fail-rate: 1.72%)
Boat - (11.36 Hours, Cost: $44, Reputation: 2.21/5, Waiting-time: 11.36 Hours, Fail-rate: 0.90%)

4 - Rome (1444km)
Plane - (1.78 Hours, Cost: $410, Reputation: 2.03/5, Waiting-time: 5.34 Hours, Fail-rate: 2.80%)
Car - (8.90 Hours, Cost: $136, Reputation: 1.60/5, Waiting-time: 0.00 Hours, Fail-rate: 3.03%)
Train - (5.34 Hours, Cost: $102, Reputation: 2.41/5, Waiting-time: 10.68 Hours, Fail-rate: 2.22%)
Boat - (14.24 Hours, Cost: $58, Reputation: 1.91/5, Waiting-time: 0.00 Hours, Fail-rate: 1.20%)

5 - Zurich (777km)
Plane - (0.97 Hours, Cost: $370, Reputation: 2.14/5, Waiting-time: 6.79 Hours, Fail-rate: 3.00%)
Car - (4.85 Hours, Cost: $123, Reputation: 1.66/5, Waiting-time: 4.85 Hours, Fail-rate: 3.68%)
Train - (2.91 Hours, Cost: $92, Reputation: 2.55/5, Waiting-time: 11.64 Hours, Fail-rate: 2.72%)
Boat - (7.76 Hours, Cost: $52, Reputation: 2.00/5, Waiting-time: 7.76 Hours, Fail-rate: 1.50%)
```

Figure 7.2: Attribute values of available transportation services

7.1.2 Modes of the game

The Ambitious-Traveler game can be played in many modes, but we have chosen to use the following two modes in our validation procedure: 1) the game is played with the intent of maximizing the distance covered given the limited resources (i.e. time and money), 2) the game is played with the intent of conforming to a set of preference weights on the selected service attributes discussed in Section 7.1.1 along with the intent of maximizing the distance traveled.

In the first mode, the player needs to maximize the distance traveled given a limited amount of time and money. There are no other constraints on the player and, therefore, the main task is to consider the trade-offs associated with the service attributes and make judicious travel selections to preserve time and money and maximize the distance traveled.

In the second mode of the game, we introduce *preference weights*. Preference weights are numerical values (real numbers between 0 and 5) assigned to service attributes based on the relative importance of the service attributes. For example, one may rate reliability as the most important service attribute, waiting time as an attribute of moderate importance and other service attributes as low importance attributes. The preference weights in this case could be: reliability - preference weight of 5; waiting-time - preference weight of 3; the remaining service attributes - preference weight of 1. This implies that the player is supposed to select services with the intent of : 1) maximizing distance as always, 2) selecting services with high reliability values, and 3) selecting services that have short queue lengths although this requirement is not as critical as service reliability. The player is not expected to conform to these requirements exactly but attempt to do so as much as possible.

7.1.3 Progression of the game

The game starts with the player in London. The player is allocated a fixed amount of money and fixed amount of travel time. The player selects a city to travel to from London and a transportation service to get to the city by weighing the service attributes described in Section 7.1.1. The selections are made dependent on which of the two modes the game is played in as described in Section 7.1.2. As the player travels through the cities, the allocated money and time depletes accordingly. The game ends as soon as one or more of the following conditions are met: 1) the player manages to visit all the cities within the allocated time and money, 2) the player runs out of the allocated time, 3) the player runs out of the allocated money.

The game is played by a set of human players and is simultaneously played using selection decisions made by the Affinity Model. The main objective is to compare the performance of the Affinity Model with the intuitive decisions of the human players. A comparable or better performance by the model will serve to validate that it can favorably match the decision making capabilities of humans in a multi-attribute service composition environment. In the following sections we describe the game as it is played by the human players and then by the Affinity Model.

7.1.4 The game played by human players

The Ambitious-Traveler game is played by a human player through a command-line based interface. The player is shown the available travel choices using the interface and articulates his/her decisions through the keyboard. The selection decisions of

the players are intuitive based on the attributes of the available choices. The players are afforded sufficient time to take all of their travel selection decisions.

As discussed earlier, the player is initially allocated a limited amount of travel time and service-purchasing money in the game. Using the time and money available the player has to travel through cities in Europe with the intent of traveling as far as possible (in Mode 1) or traveling as far as possible while conforming to specified preference weights on service attributes (in Mode 2). The player must stop when either of these resources is exhausted. Selection decisions between the available cities and transportation services are made based on the values of the service attributes discussed in the Section 7.1.1. The player needs to consider all the trade-offs and make appropriate decisions.

At every stop, the player is shown the cities available to travel to from his/her current city. The game always starts from *London* and, for each city that can be traveled to, the available transportation services are displayed. The following are shown to the player on the display: 1) list of city names and distances to the cities from the current city , 2) set of transportation services available for each city, 3) time to travel using the available transportation services, 4) cost of transit, 5) the non-functional attribute values of the transportation services – reliability, waiting-time, reputation. Figure 7.3 is a screenshot of the display showing the choices available to a customer who is currently in Berlin and needs to make his/her next selection of city and transportation services.

Only those cities and/or transportation services are shown to the player that can be traveled to given the player's available time and money. For example, with the player in Berlin shown in Figure 7.3, the amount of money left is: $459, the time left is: 1 day and 7.42 hours. The flight to Athens from Berlin costs $462 and takes 2.23 hours. This flight option to Athens is therefore not shown on the display to the player as it is more expensive than the available money. A car to Paris, on the other hand, costs just $33 but has a long queue which makes its waiting-time greater than 1 day and 7.42 hours. Therefore the car to Paris is not included in the list of available services.

The game ends either when the player has visited all the available cities or when time or money or both are exhausted. When the game ends, the distance traveled by the player and the time and money remaining are displayed.

7.1.5 The game played based on the Affinity Model

As the real player plays the Ambitious-Traveler game, simultaneously in the background the game is played based on the selections made using the Affinity Model described in Section 6.3.3. Recall that the affinity factor was defined as a 'unifying' factor that facilitated the combination of the service attribute values used for the selection of service elements.

```
You have 459.00 Dollars and 1 Days and 7.42 Hours left.
You are currently in Berlin.
You have travelled 2628 km.

Please press ENTER to continue


Please choose where you would like to go.

0 - London (930km)
Plane - (1.17 Hours, Cost: $182, Reputation: 3.36/5, Waiting-time: 3.51 Hours, Fail-rate: 2.20%)
Car - (5.85 Hours, Cost: $60, Reputation: 2.41/5, Waiting-time: 11.70 Hours, Fail-rate: 0.43%)

1 - Paris (878km)
Plane - (1.12 Hours, Cost: $112, Reputation: 4.83/5, Waiting-time: 10.08 Hours, Fail-rate: 2.40%)

3 - Barcelona (1509km)
Plane - (1.87 Hours, Cost: $282, Reputation: 2.52/5, Waiting-time: 1.87 Hours, Fail-rate: 2.80%)
Car - (9.35 Hours, Cost: $94, Reputation: 1.89/5, Waiting-time: 0.00 Hours, Fail-rate: 2.38%)
Train - (5.61 Hours, Cost: $70, Reputation: 3.07/5, Waiting-time: 11.22 Hours, Fail-rate: 1.72%)
Boat - (14.96 Hours, Cost: $40, Reputation: 2.34/5, Waiting-time: 14.96 Hours, Fail-rate: 2.34%)

5 - Zurich (668km)
Plane - (0.83 Hours, Cost: $362, Reputation: 2.17/5, Waiting-time: 0.00 Hours, Fail-rate: 3.20%)
Car - (4.15 Hours, Cost: $120, Reputation: 1.68/5, Waiting-time: 4.15 Hours, Fail-rate: 3.68%)
Train - (2.49 Hours, Cost: $90, Reputation: 2.59/5, Waiting-time: 2.49 Hours, Fail-rate: 2.72%)
Boat - (6.64 Hours, Cost: $51, Reputation: 2.03/5, Waiting-time: 13.28 Hours, Fail-rate: 2.94%)

6 - Athens (1801km)
Car - (11.15 Hours, Cost: $154, Reputation: 1.52/5, Waiting-time: 0.00 Hours, Fail-rate: 4.33%)
Train - (6.69 Hours, Cost: $115, Reputation: 2.23/5, Waiting-time: 0.00 Hours, Fail-rate: 3.22%)
Boat - (17.84 Hours, Cost: $66, Reputation: 1.78/5, Waiting-time: 0.00 Hours, Fail-rate: 3.24%)

7 - Amsterdam (576km)
Plane - (0.72 Hours, Cost: $82, Reputation: 5.00/5, Waiting-time: 5.04 Hours, Fail-rate: 3.60%)

8 - Copenhagen (361km)
Plane - (0.43 Hours, Cost: $252, Reputation: 2.79/5, Waiting-time: 2.15 Hours, Fail-rate: 3.80%)
Car - (2.15 Hours, Cost: $84, Reputation: 2.04/5, Waiting-time: 6.45 Hours, Fail-rate: 5.63%)
Train - (1.29 Hours, Cost: $63, Reputation: 3.40/5, Waiting-time: 12.90 Hours, Fail-rate: 4.22%)
Boat - (3.44 Hours, Cost: $36, Reputation: 2.55/5, Waiting-time: 3.44 Hours, Fail-rate: 3.84%)

9 - Stockholm (818km)
Plane - (1.02 Hours, Cost: $362, Reputation: 2.16/5, Waiting-time: 1.02 Hours, Fail-rate: 4.00%)
Car - (5.10 Hours, Cost: $120, Reputation: 1.68/5, Waiting-time: 5.10 Hours, Fail-rate: 6.28%)
Train - (3.06 Hours, Cost: $90, Reputation: 2.58/5, Waiting-time: 6.12 Hours, Fail-rate: 4.72%)
Boat - (8.16 Hours, Cost: $51, Reputation: 2.03/5, Waiting-time: 8.16 Hours, Fail-rate: 4.14%)
```

Figure 7.3: Service options available to player in Berlin

Affinity Model in Mode 1

The Affinity Model supports different behavior while playing the game in different modes. In Mode 1 (cf. Section 7.1.2), where the intent is to maximize the distance traveled without conforming to any constraints, the affinity factor used is shown in equation (7.1). The affinity factor combines values of all the service attributes that influence the service selection decisions. An attribute whose increasing value positively contributes to the overall service objective (in this case, maximizing distance traveled) is placed in the numerator of the affinity equation, whereas, an attribute whose decreasing value contributes to the overall service objective is placed in the denominator.

$$affinity = \frac{\{distance\}^{ND} \cdot \{reliability\}^{ND} \cdot \{reputation\}^{ND}}{\{waiting\text{-}time\}^{ND} \cdot \{cost\}^{ND}} \tag{7.1}$$

Each attribute value in the affinity equation has an exponent equal to ND which is an acronym for *Not Defined*. This is because no specific preference weight values are assigned to the attributes. As such, the favorable preference weight values need to be determined through techniques like Monte-Carlo simulation [94] described subsequently in the chapter. The affinity values are calculated for every available transportation service from the current city. A Greedy [77] approach is followed wherein the transportation service with the largest calculated affinity value is selected.

Affinity Model in Mode 2

In Mode 2 of the Ambitious-Traveler game, preference weights are assigned to the service attributes and the service selections are done conforming to these. The affinity equation is therefore modified to the one shown in equation (7.2).

$$affinity = \frac{\{reliability\}^{\alpha} \cdot \{reputation\}^{\gamma}}{\{waiting\text{-}time\}^{\beta} \cdot \{cost\}^{\delta}} \tag{7.2}$$

The service attributes are assigned one of the exponents: α, β, γ, and δ. These exponent values represent preference weights and are predefined or are set by the human player. The Greedy approach is taken to select the city and transportation service with the largest calculated affinity value.

The selections made by the real player and the Affinity Model are compared. A comparable score by the Affinity Model serves to validate its efficacy i.e., it is able to make service selections automatically that are comparable to the intuitive judgement of human players. In the following section we discuss experiments that were conducted using the Ambitious-Traveler game with the purpose of validating the Affinity Model approach to service composition.

7.2 Experiments on the Ambitious-Traveler game

The experiments on the Ambitious-Traveler game comprised a group of human participants that played the game in the two modes discussed in Section 7.1.2. The game was simultaneously played in an automated mode wherein the selections were made in accordance with the proposed service composition approach involving the Affinity Model described in Section 6.3.3. Hereafter, this automated mode of playing the game involving the Affinity Model will be referred to as a player called *affinity-man*.

7.2.1 Hypotheses

In the experiments using the Ambitious-Traveler game, the game is played by a set of human participants and simultaneously by the Affinity Model. The validation of the Affinity Model is dependent upon favorable results of the experiments. The results of the experiments can be construed as favorable if any one of the following hypotheses is satisfied for the results obtained:

Hypothesis 1: The scores (distance traveled in Mode 1 or integrated score combining degree of conformance and distance traveled in Mode 2) of affinity-man, the automated player, are *better* than the average scores of the participating human players. Better implies distance traveled is larger or integrated score is greater.

Hypothesis 2: The scores of affinity man are *comparable* to the average scores of the participating human players. By comparable we mean the scores are not better but are within 10% of the average scores of the participating human players.

7.2.2 The experimental procedure

To conduct experiments using the Ambitious-Traveler game we solicited participation from students of CMPUT 101, an introductory level course in the Department of Computing Science, University of Alberta and from graduate students (M.Sc. and Ph.D. level students) of the same department[2]. CMPUT 101 has students from various departments in the University and in fact has few students interested in majoring in Computing Science. This gave us participants with backgrounds from a wide variety of areas of study. Adding Computing Science graduate students to the participants gave us a good mix of computer and non-computer background participants. All the participants were university students who could safely be construed as 'intelligent' participants. This made it a good test for the Affinity Model in terms of its efficacy.

There were eleven participants in all. The participants volunteered to play the game in two sessions. The first session conformed to the first mode discussed in

[2]The letter of Ethics Approval is included at the end of this thesis

Section 7.1.2 where there were no preference weights placed on the service attributes. The players played the game with the sole objective of maximizing the distance traveled given a limited allocation of travel time and money.

The second session corresponded to the second mode discussed in Section 7.1.2. In this session, preference weights were assigned to the individual service attributes. The player had to play the game with the intent of conforming as closely as possible to the set preference weights while still trying to travel as far as possible. In both sessions, the time and money allocated for travel were 2 days (48 hours) and $1000 respectively. The two sessions with their respective results are described in the following sections.

7.2.3 Session 1

The first session of the experiments corresponds to Mode 1 of the Ambitious-Traveler game. In this session the eleven human players were asked to play the game with the intent of maximizing the distance traveled with a limited amount of time and money. No other constraints were imposed.

Each player played the game five times. These will hereafter be referred to as *Game 1*, *Game 2*, *Game 3*, *Game 4*, and *Game 5*. The procedure, service domain, and intent of the player in all five games were the same. The games differed from each other in terms of the 'synthetic load' on the service domain. Recall from section 7.1.1 under sub-section "Waiting-time", there was a large number (1,000) of synthetic players that played the game simultaneously with the real player. The five games differed in terms of the order and kind of synthetic players that played the game, although the number of the synthetic players was unchanged.

Each player could only see his/her results (the distance traveled) at the end of each game. The results (distance traveled) of the eleven players for Session 1 are shown in Table 7.1. The names in the first column are the code-names chosen by the participants to conceal their actual identities.

To assess the performance of the automated player, affinity-man, the 'not-defined (ND)' exponents of the Affinity Model discussed in Section 7.1.5 were calculated. This was done by running Monte-Carlo simulations [94]. The Monte-Carlo simulations comprised generating a large number (1,000) of sets of random exponents for the service attribute values in the expression for affinity in equation (7.1). Each exponent value was a real number made to vary between 0 and 5. A set of random exponents comprised of exponent values for service attributes: distance, reliability, waiting-time, reputation, and cost. The affinity values were calculated for each of these exponent combinations by substituting the exponent values in equation (7.1). Finally, using the affinity values calculated, service selections were made following the Greedy method described in section 6.4.1.

The top 11 of the generated 1,000 exponent combinations were identified on the basis of the average distance traveled by affinity-man across the five games. The distance traveled by the top 11 combinations along with the respective exponent

Table 7.1: Distance traveled by human participants.

Player name	Game 1	Game 2	Game 3	Game 4	Game 5	Player avg.
Hilronto	1505	6268	8128	7181	N/A[a]	5770.5
Redgummy	6520	9275	5369	6693	8229	7217.2
Jaxx	6298	6585	5285	5285	7250	6140.6
Decentb	4293	7040	5748	7159	6645	6177
Kavb	6579	8336	8179	9243	8596	8186.6
Dknuth	6051	7346	7681	8042	6106	7045.2
Egnaro	3103	7338	8599	8289	8343	7134.4
Grubby	6975	7469	7079	8188	6968	7335.8
Xingkai	6275	7946	9790	7593	8229	7966.6
Bond	6799	9477	7747	7553	8106	7936.4
P749	4429	4242	3993	6971	4984	4923.8
Mean	5347.91	7392.91	7054.36	7472.45	7345.6	

[a]Game 5 distance score for player *Hilronto* is not available

combinations are shown in Table 7.2. The first column is the combination number that identifies which of the generated 1,000 random combinations gave the top average distances. The next set of five columns are the preference weights for the service attributes which were obtained by assigning randomly generated numbers between 0 and 5. The next five columns in the table comprise the distance traveled in the five games using the corresponding set of preference weight values. Finally, the last column shows the average distance traveled in the five games.

Our objective is to show that these figures of distance obtained through automated selections by the Affinity Model are comparable (within a limit of 10%) or better than the selections made by the human players. Figure 7.4 shows a bar-chart depicting the comparative performances of the human players and affinity-man for each game. To set a game performance baseline for comparison purposes, each game was also played "at random" five times. A randomly played game comprised making a random selection for the city to travel to next and also a random selection of the transportation service. The results of the randomly played games are also shown on the bar-chart.

Figure 7.5 shows the average performance of individual players, human or automated, across the five games. The two bar-charts clearly indicate that the performance of affinity-man (the Affinity Model) is either better or comparable to that of the average performance of the human players. Table 7.3 shows the percentage by which the human players outperform affinity man. None of these figures exceeds 10% which validates our two hypotheses.

Table 7.2: Top performing results of affinity-man

Comb no.	rel. wt.	rep. wt.	cost wt.	wait-time wt.	dist. wt	Game 1	Game 2	Game 3	Game 4	Game 5	Comb avg.
215	2.05	1.41	1.13	2.11	3.27	4918	8980	8365	8618	6210	7418.2
804	1.47	1.02	2.03	2.51	4.98	7929	8605	6973	6304	7183	7398.8
259	4.27	1.01	1.55	4.62	3.67	4979	8744	8308	8618	5606	7251
592	3.04	1.01	2.75	3.97	4.33	7240	8744	6973	6304	6973	7246.8
685	2.74	1.03	2.81	1.96	4.77	7929	7669	6973	6210	7183	7192.8
459	3.8	1.02	2.1	2.16	3.07	7240	8261	6973	6304	6973	7150.2
403	1.54	1.07	2.34	3.59	4.82	6838	8605	6973	6304	6973	7138.6
518	1.4	1.14	1.23	1.53	2.22	5606	7669	6973	7159	7979	7077.2
102	3.86	1.09	1.6	3.19	4.34	4979	8605	6973	8618	6210	7077
553	2.51	1.04	1.9	4.67	2.9	4979	8744	6973	7224	7402	7064.4
920	1.61	1.15	4.71	4.95	4.51	6293	8261	6189	6973	7402	7023.6
Mean dist.						6266.36	8444.27	7149.64	7148.73	6917.64	

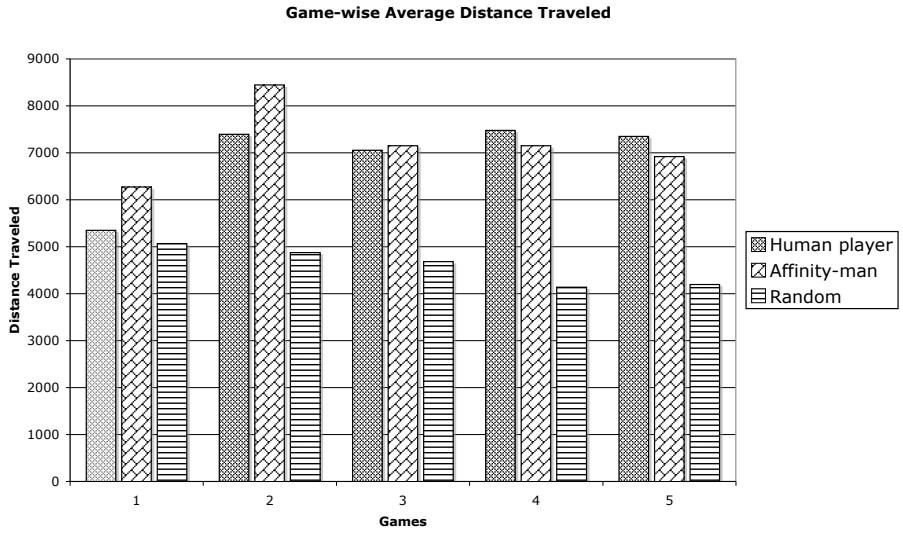


Figure 7.4: Session1: Comparison of distance traveled by human players vs. affinity-man

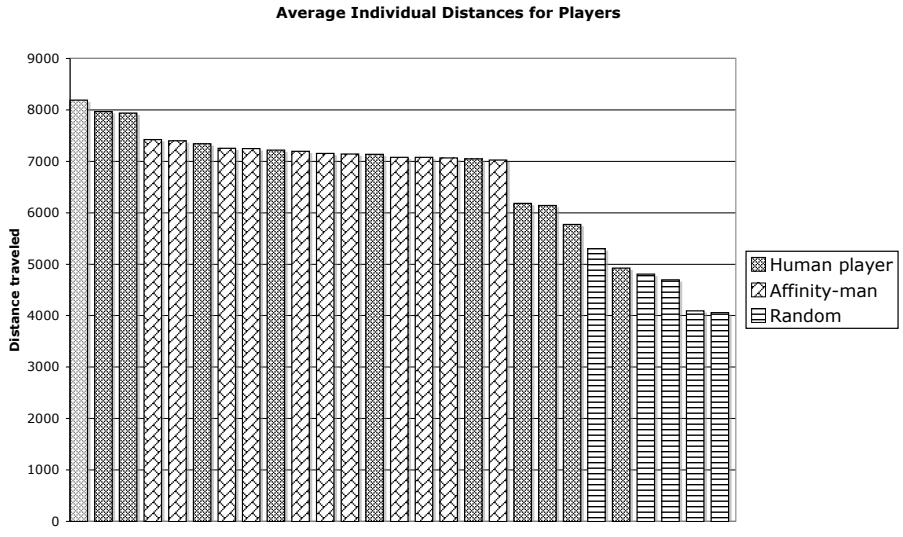


Figure 7.5: Session1: Average distance traveled

Table 7.3: Session 1: Comparison of distance traveled by human players and affinity-man.

Game no.	avg. human player dist.	avg. affinity-man dist.	Comparison
1	5347.91	6266.36	affinity-man better
2	7392.91	8444.27	affinity-man better
3	7054.36	7149.64	affinity-man better
4	7472.45	7148.73	human player better by 4.33%
5	7345.6	6917.64	human player better by 6.18%

7.2.4 Session 2

In Session 2, the players played the Ambitious-Traveler game conforming to Mode 2 described in Section 7.1.2. In this mode, the players played the game with the intent of conforming to the preference weights assigned to the service attributes while still trying to travel as far as possible. For example, the reliability attribute could be assigned a preference weight of 5 while the other service attributes could be assigned values of 1. In this case, the player playing the game would need to select high reliability services while simultaneously trying to maximize the distance traveled. The preference weights were pre-defined for the first four games and were set by the human players in the fifth game.

There were five games in Session 2. The synthetic player load in these games corresponded to those in the previous session. That is, the first game in this session had an identical composition of synthetic players as Game 1 of the first session and so on for the other games. Besides differing synthetic loads, the five games in Session 2 were also different due to varying preference weights assigned to the service attributes. Each game in Session 2 was associated with a persona based on the preference weights assigned to the service attributes as described in the following sections.

Game 1: Thrifty

The first game of Session 2 constitutes the *Thrifty* persona. While playing the game in this persona, the player behaves in a thrifty manner i.e. tries to save as much money as possible yet travel as far as possible with the money spent. The preference weights for the service attributes in the thrifty persona are: reliability: 1, waiting-time: 2, reputation: 3, cost: 5. Cost is given the largest weight while the other attributes are given smaller weights. The player needs to conform to these constraints as closely as possible while making selections of transportation services.

Game 2: Trendy

In the *Trendy* persona, the player seeks to behave like a trendy traveler paying more attention to attributes like the reputation of the service as compared to other influencing factors. The preference weights for the trendy persona used in the second game of Session 2 are: reliability: 2, waiting-time: 3, reputation: 5, cost: 2.

Game 3: Sprinter

The *Sprinter* persona places maximum weight on the waiting-time attribute. The player that takes on this persona is in a mood to 'sprint' and does not like to wait. The third game of Session 2 assigns sprinter specific preference weights to the service attributes as follows: reliability: 3, waiting-time: 5, reputation: 3, cost: 2.

Game 4: Paranoid

Game 4 constrains the player to play the game along the *Paranoid* persona. In this persona, the player makes transportation selections like a traveler who is always concerned about his/her safety and wants to use a reliable service as much as possible. The preference weights for the paranoid persona are: reliability: 5, waiting-time: 2, reputation: 1, cost: 1.

Game 5: Customized

Finally, in game 5 the player is not constrained to play the game along any persona. This persona called *Customized*, is established by the player assigning his/her own choice of preference weights to the service attributes.

The eleven human players were asked to play the five games conforming to the specified persona for each game. The players played the five games one after the other. After each game, the player's results were displayed. The results comprised: 1) distance traveled, and 2) the degree of conformance to the specified preference weights that the player was able to achieve.

Calculating the degree of conformance

The degree of conformance to specified preference weights was calculated on the basis of how close the service attribute values of the services selected by the player was to the specified weights. To calculate the conformance score the following steps are followed:
Step 1: The rank of the selected service element in terms of each service attribute is calculated as a real number value between 0 and 5. This is done by normalizing the rank of the service element in terms of the respective attribute in comparison to all the available service elements as a value between 0 and 5. For example, suppose

there are 24 service elements available with different values of reliability. The service element selected from these 24 service elements is the one whose reliability value ranks 5th out of 24. The rank of the service element in terms of reliability would then be:

$$rank = \frac{24 - (5 - 1)}{24} * 5 = 4.167$$

In general, the rank of each selected service element is calculated for each service attribute following equation (7.3).

$$rank = \frac{No.\ of\ available\ elements - (Rank\ of\ selected\ element - 1)}{No.\ of\ available\ elements} * 5 \tag{7.3}$$

Step 2: The average rank for each service attribute is calculated as the average of the ranks of all the services selected in the course of the the whole trip. This is shown in equation (7.4).

$$\forall service\ attributes : averageRank = \frac{\sum_i^{for\ each\ selection} rank_i}{total\ selections} \tag{7.4}$$

Step 3: Next, the conformance score is calculated for each persona in terms of the distance of the *averageRank* values for each service attribute from the respective preference weight allocated in that persona. This distance is multiplied with the preference weight value to put more emphasis on the service attributes that are weighted more heavily in the persona. The conformance score is calculated using equation (7.5). AR in the equation implies *averageRank*, and PW implies Preference Weight.

$$score = 1 - \frac{\sum_i^{all\ attributes} PW_i * |AR_i - PW_i|}{\sum_i^{all\ attributes} PW_i * max(|5 - PW_i|, |0 - PW_i|)} \tag{7.5}$$

For example, in the thrifty persona the preference weights are: reliability: 1, waiting-time: 2, reputation: 3, cost: 5. Assuming that for a particular game the *averageRank* values for the service attributes are calculated as: reliability = 3.4, waiting-time = 4.85, reputation = 0.55, cost = 2.6, the conformance score for the selections in that game is calculated as:

$$score = 1 - \frac{1 * |3.4 - 1| + 2 * |4.85 - 2| + 3 * |0.55 - 3| + 5 * |2.6 - 5|}{1 * 4 + 2 * 3 + 3 * 3 + 5 * 5} = 0.376$$

The best conformance score possible is 1 which implies that at each selection the service element selected exactly matched the requirements expressed by the preference weights. The worst conformance score possible is 0. The conformance scores for the human players and affinity-man are shown in Table 7.4.

Affinity-man, the automated player, played the first four games in Session 2 by substituting the specified preference weights as exponent values in the equation for

Table 7.4: Session 2: Conformance scores of players

Player name	Thrifty	Trendy	Sprinter	Paranoid	Customized	Custom. (aff.-man).
Hilronto	0.65	0.79	0.71	0.29	0.71	0.78
Redgummy	0.78	0.84	0.71	0.40	0.67	0.81
Jaxx	0.72	0.77	0.89	0.24	0.80	0.71
Decentb	0.71	0.81	0.84	0.31	0.86	0.57
Kavb	0.75	0.64	0.80	0.23	0.80	0.69
Dknuth	0.53	0.80	0.81	0.12	0.46	0.41
Egnaro	0.59	0.72	0.80	0.52	0.94	0.63
Grubby	0.68	0.84	0.91	0.14	0.78	0.82
Xingkai	0.85	0.89	0.90	0.29	0.55	0.75
Bond	0.75	0.83	0.87	0.12	0.83	0.81
P749	0.85	0.69	0.74	0.34	0.50	0.77
Mean	0.71	0.78	0.82	0.27	0.72	0.71
Affinity-man	0.72	0.80	0.83	0.58		

affinity shown in equation (7.2). The affinity values calculated were used to select services following the Greedy method described in Section 6.4.1.

For the fifth game where the persona was *customized* and the players were supposed to assign exponent values themselves, affinity-man calculated affinity using the preference weights articulated by the human players.

The conformance scores for affinity-man were found to be either better or comparable to those of the human players. Table 7.5 shows the comparison of the conformance scores achieved by affinity-man and the human players. Figure 7.6 shows the conformance scores of the human players and affinity-man on a bar-chart.

Table 7.5: Session 2: Comparison of human player and affinity-man conformance scores

Persona.	average human player conformance score	affinity-man conformance score	Comparison
Thrify	0.71	0.72	affinity-man better
Trendy	0.78	0.80	affinity-man better
Sprinter	0.82	0.83	affinity-man better
Paranoid	0.27	0.58	affinity-man better
Customized	0.72	0.71	human player better by 1.4%

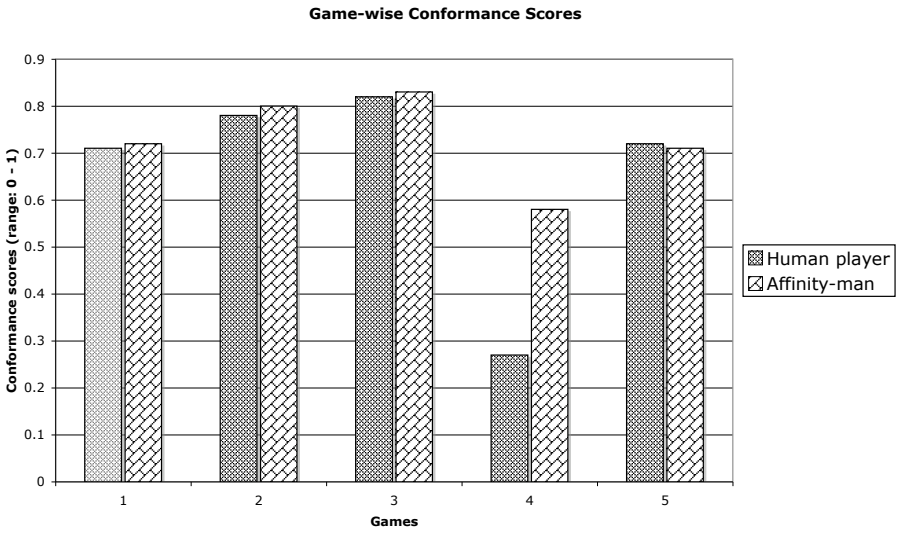


Figure 7.6: Session 2: Comparison of average conformance scores of human players vs. conformance score of affinity-man

Distance traveled by players

Besides conforming to the articulated preference weight values, the other objective of this mode of the game was to maximize the distance traveled given a limited amount of travel time and money. The distance traveled by the players in this session are shown in Table 7.6. The last row of the table shows the distance traveled by affinity-man in the first four games and the last column shows the distance traveled by affinity man in the fifth game with the *Customized* persona. The other rows and columns show the performance of the human players.

The comparison of the average distance traveled by the human players and affinity-man are shown in Table 7.7. In two of the games the human players substantially outperform affinity-man. By close inspection of the selection behavior of the human players and affinity-man we were able to establish the cause of the inferior performance of affinity-man. It was seen that towards the end of the games, when a small number of cities to travel to and transportation services were left, the human players would abandon the constraint of conforming to the preference weight values and instead make selections with the sole intention of maximizing the distance. Affinity-man, on the other hand, would continue to make selections mainly on the basis of the preference weights and put less emphasis on distance.

A bar-chart comparison of the average distance values of the human players and affinity-man is shown in Figure 7.7.

Table 7.6: Session 2: Distance traveled by players

Player name	Thrifty	Trendy	Sprinter	Paranoid	Customized	Custom. (aff.-man).
Hilronto	7223	6778	8336	7740	7978	6973
Redgummy	2120	4087	4833	8628	8605	6973
Jaxx	4185	2944	6363	3574	7095	7402
Decentb	3629	6062	8421	8429	8596	6973
Kavb	6099	6022	8599	4191	9158	6973
Dknuth	7466	1505	7095	3069	5804	1505
Egnaro	4801	1524	4976	5245	7095	6973
Grubby	3668	6493	7720	3771	4281	1505
Xingkai	2120	6344	8125	7407	8240	6099
Bond	4925	4410	7391	3069	6538	6973
P749	2120	4999	6000	6456	1505	5175
Mean	4396.91	4651.64	7078.09	5598.09	6808.63	5774.91
Affinity-man	5688	2151	7922	5856		

Table 7.7: Session 2: Comparison of the distance traveled by human players and affinity-man.

Persona.	avg. human player conformance score	affinity-man conformance score	Comparison
Thrify	4396.91	5688	affinity-man better
Trendy	4651.64	2151	human player better by 104%
Sprinter	7078.09	7922	affinity-man better
Paranoid	5598.09	5856	affinity-man better
Customized	6808.64	5774.91	human player better by 17.9%

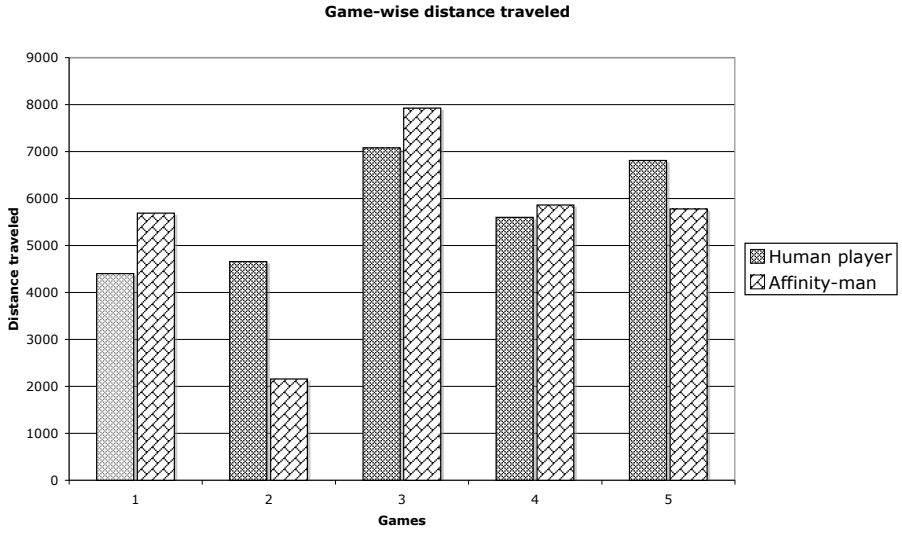


Figure 7.7: Session 2: Comparison of average distance traveled by human players vs. distance traveled by affinity-man

Table 7.8: Session 2: Integrated scores of players and affinity-man.

Player name	Thrifty	Trendy	Sprinter	Paranoid	Customized	Custom. (aff.-man).
Hilronto	4.64	5.74	2.72	4.94	6.84	6.54
Redgummy	2.47	4.03	2	6.14	7.17	6.61
Jaxx	3.33	3.16	2.57	3.16	6.45	6.65
Decentb	3.05	5.29	2.93	5.33	7.58	6.02
Kavb	4.29	5	2.91	3.28	7.82	6.32
Dknuth	4.52	2.25	2.61	2	4.86	2
Egnaro	3.38	2.14	2.16	6.04	6.76	6.17
Grubby	3.01	5.63	2.88	2.4	4.54	3
Xingkai	2.6	5.61	2.95	4.83	6.67	5.88
Bond	3.74	4.23	2.75	2	6.15	6.61
P749	2.6	4.4	2.28	4.94	2.09	5.32
Mean	3.42	4.32	2.61	4.1	6.01	5.56
Affinity-man	4.03	2.69	2.81	6.74		

Table 7.9: Session 2: Comparison of the integrated scores of human players and affinity-man.

Persona.	avg. human player conformance score	affinity-man conformance score	Comparison
Thrify	3.42	4.03	affinity-man better
Trendy	4.32	2.69	human player better by 61%
Sprinter	2.61	2.81	affinity-man better
Paranoid	4.1	6.74	affinity-man better
Customized	6.01	5.56	human player better by 8%

Integrated score

The ultimate objective of this session was to travel as far as possible while conforming to the specified preference weights on the service attributes. To assess this combined objective, a normalized integrated score combining the degree of conformance to preference weights and distance traveled was developed wherein 60% weight was put on the degree of conformance to preference weights and a 40% weight was put on the distance traveled. The integrated scores for the human players and affinity-man are shown in Table 7.8. A large integrated score indicates a good performance.

Table 7.9 shows the comparative performance of the human players and affinity-man in terms of the integrated scores and a bar-chart representation of the same is shown in Figure 7.8.

The performance of affinity-man in terms of the combined objective as indicated by the integrated score is shown to be better or comparable to that of the human players in 4 our of 5 cases (Table 7.9). The case where affinity-man was not able to perform well was due to the end-game advantage, explained earlier, that human players have which helps them cover a larger relative distance. For the most part of this session, however, the performance of affinity-man validates our two hypotheses and demonstrates the efficacy of the Affinity Model service composition approach.

7.3 Follow-up experiments

We conducted a second set of experiments using the Ambitious-traveler game after completing the first set. These experiments were similar in procedure to the first set with the following variations: 1) the players at this stage were more experienced as they were playing the same set of games a second time; and 2) the players at this stage were given feedback on the performance of the other players after every game, with the intention of incorporating peer-pressure as a motivating factor. These variations were relevant only to the human players and did not affect the performance

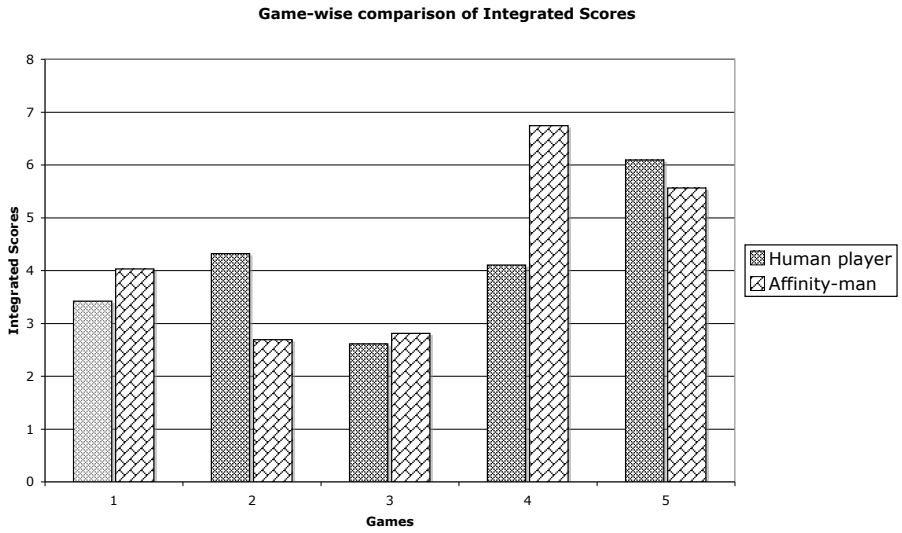


Figure 7.8: Session 2: Comparison of average integrated scores of human players vs. integrated scores of affinity-man

of the automated player, affinity-man. Nonetheless, for Session 1 of the follow-up experiments a new set of Monte-Carlo simulations were run for affinity-man and a new set of top performing preference weight combinations were identified.

The objective of conducting the follow-up experiments was to assess the learning behavior of players. To observe which aspects of the game the players were most likely to pick up quickly and improve. The aim in future work would be to incorporate some of the learning behavior of human players into the automated player and in effect develop an adaptable dynamic service composition model.

Session 1, Part 2

The follow-up experiments for the first session were conducted immediately after Session 1 of the first part. The participants were given a small break after which they played the set of five games again. Unlike the first part, the players would pause after completing each game. The players waited until all the other players were done and then they were given feedback on the performance of others. The feedback comprised, without disclosing the player names, the best performance in the game, the game average, and the worst performance. The players therefore played the subsequent game with an idea of their position relative to others in the earlier games. The distance traveled by the human players in the five games are shown in Table 7.10.

Table 7.10: Distance traveled by human participants in the follow-up experiments

Player name	Game 1	Game 2	Game 3	Game 4	Game 5	Player avg.
Hilronto	7425	8425	7900	9716	8066	8306.4
Redgummy	6284	9535	9790	8599	8596	8560.8
Jaxx	7139	8917	5822	9535	8596	8001.8
Decentb	7949	8640	8068	9535	8208	8480
Kavb	8609	8557	4155	9716	7664	7740.2
Dknuth	7171	8599	7099	8599	9716	8236.8
Egnaro	6869	9052	8980	9535	9205	8728.2
Grubby	7426	8104	7902	8853	6645	7786
Xingkai	7109	8023	7902	8557	8640	8046.2
Bond	7466	8981	8462	7517	8636	8212.4
P749	4191	8115	4908	5424	7418	6011.2
Mean	7058	8631.64	7362.55	8689.64	8308.18	

The distance traveled by top performing preference weight combinations from another set of 1,000 randomly generated preference weights are shown in Table 7.11. The table shows the preference weights for the service attributes, the distance traveled using the preference weights over the five games and the average distance traveled.

There is approximately a 25% improvement on average in the distance traveled by human players in the follow-up experiments (Table 7.10) when compared to the results of the first set of experiments (Table 7.1). This reflects on the ability of human players to learn and quickly adapt their game to improve their scores. The percentage improvement in the distance traveled for individual players is shown in Table 7.12.

Affinity-man on the other hand, due to a lack of learning mechanism remains relatively consistent through the two sets of experiments. There is in fact a slight decrement in the average distance traveled by the top performing preference weight combinations of affinity-man over the two sets of experiments. The percentage decrement over the two sets of experiments (Table 7.11 and Table 7.2) on an average is 0.26% .

A comparison of the average performances of human players and affinity-man over the two sets of experiments is shown in the bar-chart representation in Figure 7.9.

Session 2, Part 2

The follow-up experiments for Session 2, which involved conformance to assigned preference weight values to service attributes, were conducted a few days later. In fact, due to unavailability of the human participants the experiments were con-

Table 7.11: Top performing results of affinity-man from a repeat simulation

Comb no.	rel. wt.	rep. wt.	cost wt.	wait-time wt.	dist. wt	Game 1	Game 2	Game 3	Game 4	Game 5	Comb avg.
319	3.92	1.21	1.24	1.91	4.42	5606	8093	8365	8365	6210	7327.8
509	2.81	1.03	1.36	2.65	2.88	4979	8744	8308	8618	5606	7251
99	4.55	1.02	2.18	2.76	3.11	7240	8744	6973	6304	6973	7246.8
141	2.08	1.55	1.33	3.87	4.83	4918	8980	8365	8618	5175	7211.2
507	3.71	1.13	2.12	2.59	4.39	7240	7669	6973	7159	6973	7202.8
551	3.03	1.54	1.56	3.38	4.91	4918	7669	8365	8618	6210	7156
161	1.25	1.47	1.35	2.5	4.66	5606	7233	8365	8365	6210	7155.8
712	2.78	1.02	2.85	2.87	4.32	7240	8261	6973	6304	6973	7150.2
911	4.81	1.09	2.21	4.07	4.35	4979	8744	6973	6304	8474	7094.8
27	1.3	1.03	3.96	4.49	3.98	6293	8261	6189	6973	7402	7023.6
930	1.89	1.11	2.15	4.56	3.31	4979	8744	6973	6973	7402	7014.2
Mean dist.						5818	8285.64	7529.27	7509.18	6691.64	

Table 7.12: Percentage improvement in distance traveled by human players in follow-up experiments.

Player name	Game 1	Game 2	Game 3	Game 4	Game 5	Player avg.
Hilronto	393.36%	34.41%	-2.81%	35.3%	N/A[a]	115.07%
Redgummy	-3.62%	2.8%	82.34%	28.48%	4.46%	22.89%
Jaxx	13.35%	35.41%	10.16%	80.42%	18.57%	31.58%
Decentb	85.16%	22.73%	40.36%	33.19%	23.52%	40.99%
Kavb	30.86%	2.65%	-49.2%	5.12%	-10.84%	-4.28%
Dknuth	18.51%	17.06%	-7.58%	6.93%	59.12%	18.81%
Egnaro	121.37%	23.36%	4.43%	15.03%	10.33%	34.9%
Grubby	6.47%	8.5%	11.63%	8.12%	-4.64%	6.02%
Xingkai	13.29%	0.97%	-19.28%	12.7%	4.99%	2.53%
Bond	9.81%	-5.23%	9.23%	-0.48%	6.54%	3.97%
P749	-5.37%	91.3%	22.92%	-22.19%	48.84%	27.1%
Mean	62.11%	21.27%	9.29%	18.42%	16.09%	25.44%

[a] the distance score for player *Hilronto* in the first set of experiments is not available

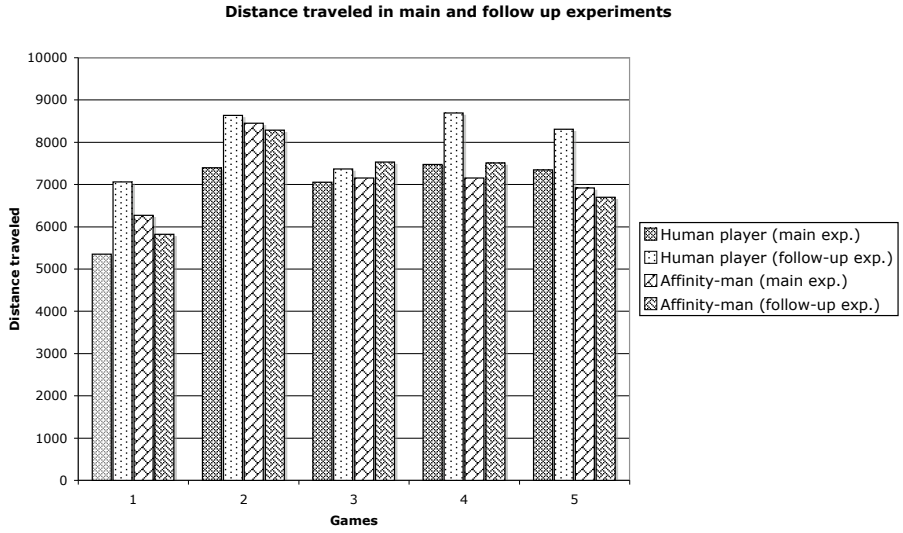


Figure 7.9: Comparison of the distance traveled by the players in the main and the follow-up experiments

Table 7.13: Conformance scores of players in the follow-up experiments.

Player name	Thrifty	Trendy	Sprinter	Paranoid	Customized	Custom. (aff.-man).
Hilronto	0.71	0.78	0.78	0.12	0.66	0.81
Redgummy	0.41	0.81	0.71	0.23	0.75	0.74
Jaxx	0.53	0.75	nil	0.29	0.88	0.57
Decentb	0.71	0.82	0.84	0.29	0.85	0.66
Kavb[a]	N/A	N/A	N/A	N/A	N/A	N/A
Dknuth	0.5	0.78	0.8	0.35	0.82	0.57
Egnaro	0.36	0.54	0.74	0.44	0.89	0.63
Grubby	0.46	0.8	0.85	0.29	0.69	0.87
Xingkai	0.65	0.7	0.86	0.26	0.76	0.71
Bond	0.81	0.81	0.75	0.23	0.89	0.73
P749[b]	N/A	N/A	N/A	N/A	N/A	N/A
Mean	0.57	0.75	0.79	0.27	0.79	0.7
Affinity-man	0.71	0.8	0.83	0.58		

[a]Player *Kavb* chose not to participate
[b]Player *P749* chose not to participate

ducted at different dates and time for different participants and online for a few participants.[3] Two participants of the earlier experiments chose not to participate in this part of the follow-up experiments.

This part of the experiment was also different from the first set in that the players had the experience of playing the game and the players were given feedback on the performance of other players after each game. The feedback comprised conveying the best, average, and worst performance until that point.

Apart from the *Customized* persona, no new experiments were conducted for affinity-man in this set of experiments. The conformance scores of the human players for this part of the experiments are shown in Table 7.13. Figure 7.10 shows the comparative performance of the human players in both set of experiments in terms of conformance scores. The conformance scores of affinity-man are also shown. Surprisingly there is a 'dip' of 3.9% in the performance of human players on average over the five games in spite of the experience and feedback information (Tables 7.4 and 7.13 show the conformance scores for the two sets of experiments respectively). This dip in the player performance could be attributed to the time gap between the main and follow-up experiment sessions as well as the fact that

[3]The follow-up experiments for this session had to be conducted in this way because on the original date of experiments there was a server failure due to which the remaining part of the experiments had to be abandoned. Furthermore, the participants had conflicting schedules on subsequent dates owing to which experiments had to be conducted at different dates for individual participants and in some cases online via skype.

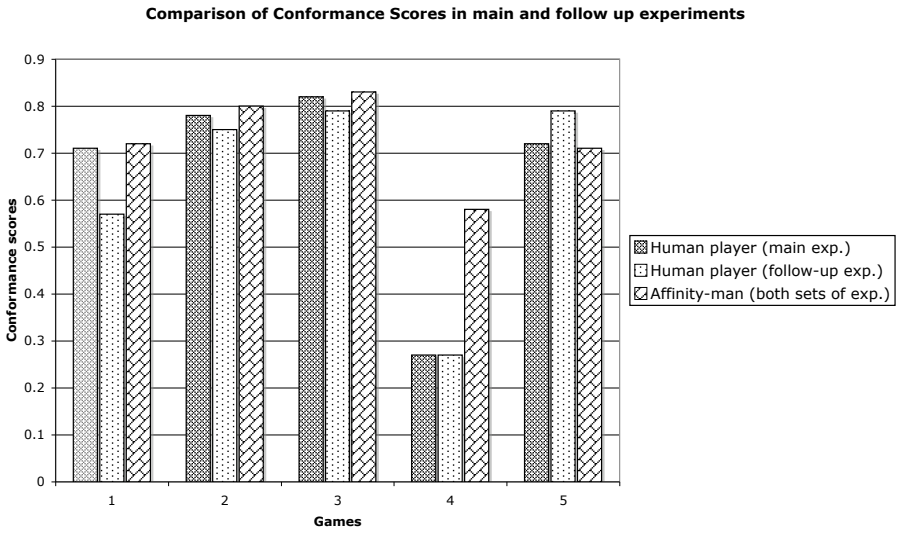


Figure 7.10: Comparison of the conformance scores of players in the main and the follow-up experiments

the follow-up experiments were conducted online for some of the players and those players could have been in locations with other distracting factors.

The distance traveled by the human players in the follow-up experiments of Session 2 are shown in Table 7.14. Figure 7.11 shows the comparison between the first experiment session and the follow-up experiments. There is a 2.83% improvement in the performance of the human players in the follow-up experiments (Tables 7.6 and 7.14 show the distance traveled in the two sets of experiments respectively). This is an insignificant improvement and could indicate that the human players did not learn much in this part of the game, although it could also be attributed to the time gap between the experiments and other distractions.

Finally, the integrated scores combining the distance traveled and conformance scores, were also mixed with slight improvements in the *Thrifty*, *Paranoid*, and *Customized* personae. *Trendy*, and *Sprinter* undergo slight decrements from the main session. On an average there is a minuscule 0.83% improvement in the integrated scores over the two sets of experiments (Tables 7.8 and 7.15).

The integrated scores comprising a 60% weight to conformance and a 40% weight to distance are shown in Table 7.15. A comparative bar-chart for the integrated scores is shown in Figure 7.12.

Table 7.14: Distance traveled by players in the follow-up experiments of the second session.

Player name	Thrifty	Trendy	Sprinter	Paranoid	Customized	Custom. (aff.-man).
Hilronto	4279	2151	3569	3069	6190	6973
Redgummy	7095	6062	4833	4725	5125	6190
Jaxx	5326	3297	8853	7722	7095	5606
Decentb	4279	5297	8853	7200	9158	5691
Kavb[a]	N/A	N/A	N/A	N/A	N/A	N/A
Dknuth	7099	2151	8599	5195	7968	5606
Egnaro	6833	7221	6072	9044	8596	6973
Grubby	5499	1505	8599	6475	7250	1505
Xingkai	7233	7524	7680	3877	8042	6996
Bond	356	2234	5850	4191	4801	6973
P749[b]	N/A	N/A	N/A	N/A	N/A	N/A
Mean	5333.22	4160.22	6989.78	5722	7136.11	5834.78
Affinity-man	5688	2151	7922	5856		

[a]Player *Kavb* chose not to participate
[b]Player *P749* chose not to participate

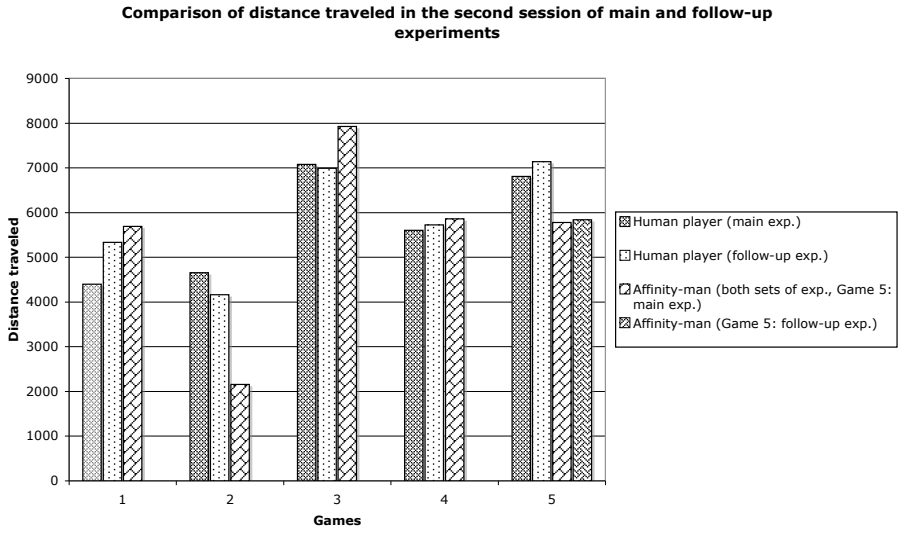


Figure 7.11: Comparison of the distance traveled by players in session 2 of the main and follow-up experiments

Table 7.15: Integrated scores of players and affinity-man in the follow-up experiments.

Player name	Thrifty	Trendy	Sprinter	Paranoid	Customized	Custom. (aff.-man).
Hilronto	3.35	2.65	1.84	2.01	5.55	6.6
Redgummy	4.12	5.29	2	3.44	5.03	5.92
Jaxx	3.5	3.36	1.83	4.96	6.63	5.12
Decentb	3.35	4.8	3.03	4.78	7.93	5.38
Kavb[a]	N/A	N/A	N/A	N/A	N/A	N/A
Dknuth	4.29	2.65	2.91	4.59	7.08	5.11
Egnaro	3.91	5.64	2.3	6.59	7.64	6.18
Grubby	3.46	2.25	2.97	4.56	6.31	3.11
Xingkai	4.63	6.1	2.79	3.43	6.99	6.37
Bond	1.69	2.75	2.27	3.26	5.14	6.42
P749[b]	N/A	N/A	N/A	N/A	N/A	N/A
Mean	3.59	3.94	2.44	4.18	6.48	5.59
Affinity-man	4.03	2.69	2.81	6.74		

[a]Player *Kavb* chose not to participate
[b]Player *P749* chose not to participate

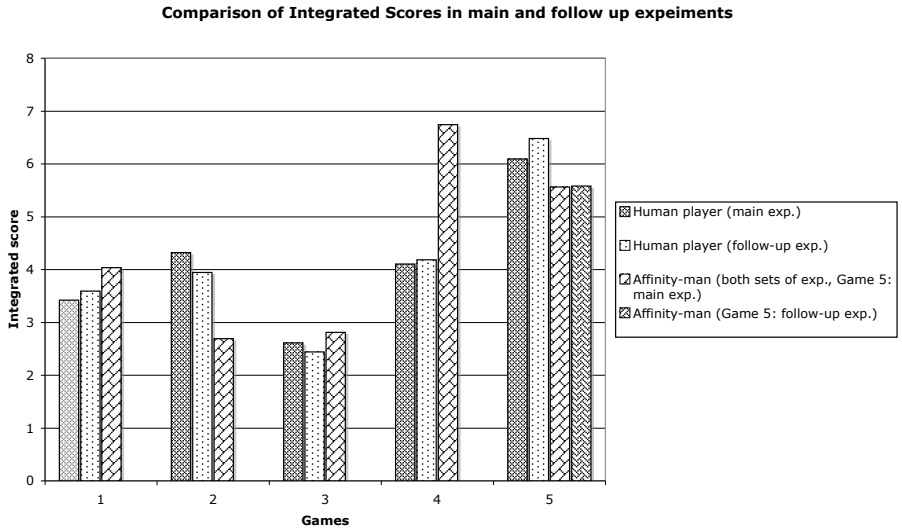


Figure 7.12: Comparison of the integrated scores of players in the main and the follow-up experiments

7.4 Summary

The Ambitious-Traveler game, which was conceived and developed by us, was described in this chapter as our approach to validate the Affinity Model of dynamic service composition. The game was found to be a simple and practicable solution to simulate an actual service composition scenario on which to run experiments to validate the model.

The experiments comprised a group of human participants who volunteered to play the Ambitious-Traveler game. The game was simultaneously played by the Affinity Model. The validation procedure involved comparing the performance of the human players with that of the Affinity Model. A better or comparable performance by the model with the human players served to validate its efficacy.

The results of the experiments did indicate that affinity-man's performance was better in 6 games and comparable in 3 games to that of the human players making it 9 out of 10 (90%) games and thus validating the efficacy of the Affinity Model.

In the assessment of distance traveled in Session 2 where the main goal was conformance to the preference weight constraints, the human players did outperform affinity-man in two of the games. By closely inspecting the game play, the reason for this was found to be the 'end-game' (the last part of the game with very few cities left) behavior of the human players. In the end-game portion the human players had a tendency to abandon the constraint of conforming to the preference weights and concentrate only on maximizing the distance traveled. Affinity-man, on the other hand, continued to conform to the constraints in the end-game portion.

Finally, follow-up experiments were conducted to examine the learning behavior of human players and affinity-man. Human players showed a healthy 25% average improvement in the distance traveled in the first session. The improvement of the human players in the second session of the follow-up experiments was however insignificant: conformance score: -3.89%, distance traveled: 2.83%, and integrated score: 0.83%. The lack of improvement in the second session could be attributed to the fact that the experiments were conducted after a time lag of a few days and the fact that the experiments were conducted online for some of the players and there could have been other distractions influencing their performance. On the other hand, affinity-man due to the lack of a learning mechanism did not show much improvement in the first session of the follow-up experiments. In the second session, apart from the *Customized* persona, experiments were not conducted for affinity-man.

Chapter 8

Conclusion and future work

Chapter 1 provided a set of research objectives that we wanted to fulfill through the research for this thesis. In Section 8.1 we re-state each objective and explain how our research accomplished these objectives. The procedure followed to validate our results is discussed in Section 8.2. Finally, Section 8.3, describes some ongoing research related to this thesis as well as related research we hope to pursue in future.

8.1 Fulfillment of research objectives

In this section, we re-state our research objectives and describe how the work in the thesis helps in achieving the objectives.

Objective: *to devise a model that facilitates the composition of service elements dynamically.*

The Affinity Model facilitates the run-time selection and composition of service elements. The affinity value is dynamic in the sense that it is calculated using the most current service attribute values of the service elements. The Greedy algorithm selects the service element with the largest affinity value during the composition process of the Affinity Model. Therefore the development of the Affinity Model satisfies our first objective.

Objective: *to devise and validate novel techniques for dynamic service composition based on non-functional attributes.*
This objective was fulfilled by developing several new techniques for service composition based on three non-functional attributes: reliability, waiting-time, and reputation.

In Chapter 3, we devised a technique to assess the reliability of a service element in a dynamic composition environment where service elements are continuously interacting with one another. The technique is novel in its incorporation of the influence of the interacting service elements in the reliability values of individual service elements. The validation of this approach is given at the end of Chapter 3.

Two techniques were devised for the assessment of waiting-time for service requests at service elements in Chapter 4. The first approach adapts concepts from Queueing-Theory to arrive at an expression for the average waiting-time in a dynamic environment. The approach is validated by running simulations with a large number of requests and applying the devised technique. The second approach to waiting-time assessment utilizes current data on the number of requests waiting and the completion rate of the service element to come up with good approximations for waiting-time values. A simple ratio of the number of waiting requests to the completion rate values is used as a basis for service element selection. We use an inductive argument to show that this approach is optimal in a dynamic service environment.

To assess service reputation, we used feedback from customers who have previously used the service. The issue with feedback is that it is strongly influenced by factors that are variable (e.g. city, cost, etc. for a hotel service) and this makes it difficult to assess reputation. The technique devised in Chapter 5 utilizes multiple regression to compensate for variations in the influencing factors and allows for more accurate assessment of service reputation from customer feedback.

Objective: *to incorporate multiple service attributes simultaneously in the process of dynamic service composition.*
The Affinity Model also fulfills this objective. The simplicity of the Affinity Model allows a new service attribute to be introduced into the composition process and/or a service attribute to be removed from the process dynamically. This is made simple because of the structure of the affinity equation shown in equation (8.1).

$$affinity = \frac{\{attribute\ value_a\}^{\alpha} \cdot \{attribute\ value_b\}^{\beta} \cdots \{attribute\ value_k\}^{\gamma}}{\{attribute\ value_m\}^{\delta} \cdot \{attribute\ value_n\}^{\epsilon} \cdots \{attribute\ value_x\}^{\zeta}} \tag{8.1}$$

A value for a new service attribute is included in the equation as part of the numerator if a large value of the attribute is desirable or as part of the denominator if a small value is desirable. The emphasis on the attribute is adjusted using an exponent on the service attribute value. A service attribute therefore can be assigned greater influence in the selection decisions by simply increasing its exponent value. Conversely, an attribute can be easily de-emphasized by reducing the value of its exponent or effectively removed by setting its exponent to a value of 0.

In this way a large number of service attributes can be used to simultaneously influence the value of affinity which is used to determine selections of service elements in a dynamic composition environment.

Objective: *to devise a procedure that facilitates direct involvement of customer in the process of service composition such that the service composition formed closely conforms to the preferences of the customer.*
In the thesis we develop a systematic procedure to ensure direct involvement of the

customer in the process of service composition. As described in Section 6.2, the customers are first requested to articulate their preference of each service attribute in terms of a *preference weight*, a value between 0 and 5. Customers who are unable to assign a number to their preferences are assisted in doing so through the *Hypothetical Equivalents and Inequivalents Method* (developed in Section 6.2.1). Once the preference weights of the customers are obtained, these are included in the affinity equation as exponents of the service attributes shown as α, β, γ etc. in equation (8.1). For example a customer who expresses a high preference for $attribute_a$ in equation (8.1) could assign a preference weight of 5 to α.

The affinity value now is directly influenced by the preferences of the customer. As explained earlier, the affinity equation plays a direct role in selecting service elements for the composition and thus the customer preferences are incorporated in the service composition process.

8.2 Validation of the approach

It can be seen that the Affinity Model plays a central role in fulfilling the objectives stated in this thesis. It is imperative therefore that the efficacy of the Affinity Model be validated. To do this, we simulated a service composition scenario in the form of a game called the *Ambitious-Traveler*, described in detail in Chapter 7. The Ambitious-Traveler game is a single player game in which a player travels between cities in Europe by selecting from a set of transportation services for every pair of cities. The transportation service selections are made on the basis of the service attribute values of the transportation services in a manner that is analogous to the more general use of service attribute values for making service selections in a real service composition environment. The aim of the player in the game is to travel as far as possible between the cities given a limited amount of travel time and money. The player has to select transportation services judiciously with the correct balance of service attribute values to minimize the time and money spent and maximize the distance covered.

We conducted several experiments on the Ambitious-Traveler game with the objective of validating the Affinity Model. The experiments involved a set of human participants who volunteered to play the game. Simultaneously, the game was also played in an automated manner based on transportation selections made by the Affinity Model. A comparable performance by the Affinity Model to that of the human players in the game served to validate the contention that the Affinity Model was able to make service selection decisions that were comparable to the intuitive judgement of humans. By comparable we mean, the Affinity Model's performance is either better or within 10% of the average performance of the human players.

The game was played in two modes in our experiments: 1) in the first mode, the aim of the players was to travel as far as possible with the allocated time and money; and 2) in the second mode, preference weights were attached to the service attributes of the transportation services and the players made service selections with

the constraint of conforming as closely as possible to these preferences. In addition to this, the players also aimed at traveling as far as possible just like in Mode 1. The results of the game comprised of the distance traveled expressed in kilometers, and a conformance score (second mode only) expressed as a real number value between 0 and 1 based on the degree of conformance to the attribute preference constraints.

The performance of the Affinity Model was found to be superior in 6 out of 10 games and comparable to that of the human players in 3 out of 10 games making the results favorable in 9 out of 10 games. The results of the experiments therefore validated the overall efficacy of the Affinity Model.

We also conducted a set of follow-up experiments which were largely similar to the first set of experiments. There were, however, two points of difference from the first set of experiments: first, the players were given feedback after every game on the relative performance of other players, and second, the players were now experienced players having completed the first set of experiments. The follow-up experiments were conducted with the intent of exploring the learning behavior and reaction to peer pressure of players. In these follow-up experiments, the limitation of the Affinity Model in terms of its inability to learn from experience and adapt accordingly was exposed. Human players on the other hand were relatively more adept at this and this was reflected in the results of the follow-up experiments.

8.3 Future research directions

Currently we are pursuing or hope to pursue in the near future several research directions related to the thesis.

8.3.1 Assessment techniques for other service attributes

In this thesis, we developed techniques for the assessment of three non-functional attributes: reliability, waiting-time, and reputation that were incorporated into the Affinity Model. Future research could be directed towards the development of techniques for assessment of other non-functional service attributes.

One significant service attribute for which we are currently exploring assessment techniques is cost. Cost, as discussed in Section 2.1, is a 'special' service attribute that can be considered both functional and non-functional in nature. It usually plays a major part in making service selection decisions.

One direction of research that we are exploring for assessing cost entails the use of the concepts of 'demand and supply' [95]. Cost is a highly dynamic service attribute and its value undergoes rapid fluctuation with the variations in demand and supply. The challenge is coming up with an appropriate mapping between the demand and supply gap and assigned costs.

Another direction of research for the assessment of the cost attribute is the use of the concepts of *Hedonic Regression* [96]. Hedonic regression, traditionally used in pricing in the real estate sector, assigns a parameter to all other attributes of the

service and an equation for cost is derived using these parameters. This is done on the basis of available historical data. Once an equation is established, the current attribute values are substituted for the parameters in the equation and the cost is calculated.

8.3.2 The Affinity Model as a simulation tool to assist in SLA management

Research in this direction would allow the Affinity Model to be used to assess 'co-value' to both service customers and providers. The Affinity Model in its current form provides value to service customers by assuring a service composition that conforms to their articulated preferences. We are exploring a utility of the Affinity Model that would make it a useful simulation tool for service providers to better negotiate Service Level Agreements (SLAs) with customers and hence make better deals.

In a dynamic service composition environment it is difficult for service providers to assess the capabilities of composite applications formed especially in terms of their ability to meet the requirements articulated by customers. To overcome this uncertainty, providers can run simulations of their service environment using the Affinity Model. This involves substituting the requirements of the service customers in the form of the preference weight exponents in the affinity equation and substituting the capabilities of the available service elements in the form of service attribute values. When new customers arrive, their requirements are expressed by specifying parameters in the affinity equation and a large number of simulation runs are conducted. Each simulation run involves the formation of a service composition dynamically using the Greedy approach of the Affinity Model. If the service compositions formed consistently meet the requirements of the new customer an SLA can be negotiated with the customer. Otherwise, the customer is requested to 'tone down' his requirements, or alternatively, the available resources needed for the service elements are increased to meet the customer requirements.

This research is still at a nascent stage and a lot of provider related issues need to be investigated. However, the approach gives some assistance to service providers in negotiating SLAs with customers in an otherwise unpredictable environment.

8.3.3 Adaptive capability in the Affinity Model

The experiments conducted on the Ambitious-Traveler game indicate definite potential in the Affinity Model to come up with compositions comparable or even better than those achieved by human volunteers in conforming to the articulated requirements. However, as the follow-up experiments demonstrate, this advantage is reduced once the human volunteers gain experience and are given feedback. The humans are quick to 'learn' and improve their performance. The Affinity Model however, in its current form, does not have the capability to learn from experience

and improve its performance.

We plan to pursue research in the near future towards making the Affinity Model more adaptive. Such a model would incorporate an optimization strategy that would learn from historical selection decisions and their implications and improve accordingly. One possible approach is the use of Genetic Algorithms [8] to learn from historical selection results and improve.

8.3.4 Analysis of the performance of human players in the follow-up section of experiments

The set of experiments that we conducted as a follow-up to the original set of experiments gave results that deserve more detailed analysis. Recall that in the follow-up experiments (Section 7.3), the players of the Ambitious-Traveler game were provided with feedback on the performance of their participating peers and the players also had the advantage of experience (having already played the same set of games once). The performance of the human players was mixed although there was distinct improvement in terms of the distance covered in the first session where it was approximately 25% on average. This is a significant improvement and we plan to conduct more systematic analysis of the results. We hope to uncover interesting correlations between the experience and peer-pressure in the performance of human players.

8.3.5 Enhancements to the experimental research platform: Ambitious-Traveler game

There are a few enhancements to the Ambitious-Traveler game in its role as an experimental research platform that we are in the process of making or plan to in future.

Web enabled game with user-friendly GUI

The Ambitious-Traveler experiments in the thesis were conducted using a command-line version of the game where the progress of the game, transportation choices available, and results were displayed as simple text to the players. Through informal feedback from the participating players, we gathered that the command-line interface of the game sometimes made it difficult for them to make appropriate transportation choices. A visually appealing interface with a map showing the players' progress would perhaps make it easier to play the game and also "less boring".

The other important feedback was the need to make the Ambitious-Traveler game web enabled and be accessible over the Internet. This would allow much wider participation of the game spread over a longer period of time. Also, the environment in which the experiments are conducted would not be as 'closed' as it was with most of the experiments of the thesis.

Concurrent to the work of this thesis we supported the task of developing a web-enabled version of the Ambitious-Traveler game with a user-friendly GUI by a group of final year undergraduate students as a course project for their CMPUT 401 course. The students did a commendable job and the new version of the Ambitious-Traveler game is functional. This version is web-enabled and has a much more appealing interface. The game has the capability of showing the players their progress through the game as well as that of the automated player, affinity-man. A screen-shot of the new version of the Ambitious-Traveler game is shown in Figure 8.1. The screen-shot shows the map of Europe and the location of cities that can be traveled to. The transportation services available are shown at the bottom and the progress of the human player and affinity-man are shown on the right of the map.

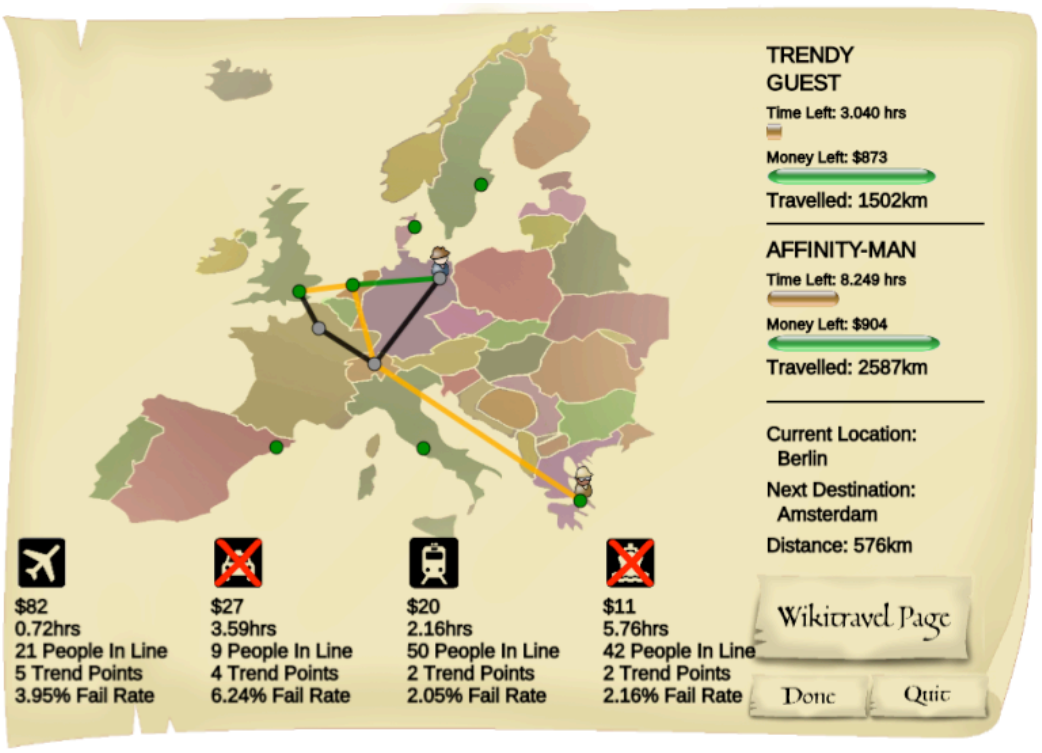


Figure 8.1: Screen-shot of the new version of the Ambitious-Traveler game.

We plan to conduct more experiments with the game using the new version. We would like to explore the variations in the results, if any, due to the more friendly interface and due to the convenience of running the game in more familiar surroundings. Wider participation which is possible by making the game accessible over the Internet would provide more data to more thoroughly examine and validate our model.

Including the service-provider in the game

Another future enhancement to the Ambitious-Traveler game that we are planning would involve a set of players who play the game as service providers in addition

to the players who play the game as customers. These provider players would be responsible for setting up transportation services that would be used by customers to move from one city to another. The providers could adjust the cost of the services to ensure a comfortable profit margin while keeping up with competition from other service providers. The service attributes other than cost of the transportation services could also be adjusted dynamically depending upon the resources available and the level of demand. The provider would benefit immensely from the game by being in a position to try out various marketing strategies in a synthetic set up and getting an idea of the level of service demand to expect.

We are hopeful that this enhancement to the game would eventually lead to the development of the game as an application for a mapping service like Google maps [97]. Prospective service providers could set up fictitious services at different locations on the map. Prospective customers could use such services, provide feedback, choose between competing services, and much more. A virtual-world travel-service application could be set up with prospective customers and providers indulging in transactions and activities that are similar to a real situation. All this would give both the customer and provider a 'feel' for the situation and ultimately enable them to make much better business decisions.

An enhancement to the Ambitious-Traveler game in this direction would also enable a 'co-value' experimental platform for service compositions to both customers and providers as discussed in Section 8.3.2.

8.3.6 Limitations of the assumptions in the thesis

In Section 2.3.3 we enumerated several assumptions that we make on the service composition environment in this thesis. In this sub-section, we address the limitations of some of these assumptions and suggest future research directions that could be taken to overcome these restrictions.

The first assumption on the service composition environment is that the service elements that partake in the composition are available and need not be discovered. Service discovery is a challenging task especially in a dynamic environment where the attributes associated with service elements are constantly updated. To address this issue future research could be directed towards the development of service registries with provisions for dynamic updates of key service attributes for the registered services. These updates could include changes in the availability status of service elements, location, and other attributes of the service elements explored in this thesis. The easy availability of current information on service elements will make dynamic discovery possible.

Another assumption that we make is regarding the underlying technology used in making service compositions. Our discussions are at a high level of abstraction and we do not examine the available technologies for putting the procedures into practice. Future research in this direction could be focussed on the development of tools and techniques that extend accepted static service composition standards

like BPEL [3] to support dynamic compositions. The extensions could include provisions for accommodating run-time decisions on the composition structure and formation of messages to be passed between the service elements.

Finally, we assume the absence of governance and legal issues in our composition environment. This is an unrealistic assumption and needs to be addressed in future research. It is a challenge to devise governance and legal structures in a dynamic environment which requires quick resolution of contractual issues between service element providers in a service composition environment. Research in this direction could be towards developing agile techniques for governance using more open-ended reciprocal agreements between service providing organizations.

Bibliography

[1] A. Metzger. *Service Composition and Co-ordination*. http://www.s-cube-network.eu/

[2] B. Srivastava, and J. Koehler. Web Service Composition - Current Solutions and Open Problems. In Proceedings of the *ICAPS Workshop on Planning of Web Services*.

[3] J. Mendling. Business process execution language for web services (BPEL). In Proceedings of the *International Workshop on Web Services Modeling and Testing (WS-MaTe)*, 2006, pp. 51-63.

[4] http://www.w3.org/TR/wscl10/

[5] A. Bucchiarone, and S. Gnesi. A survey on services composition languages and models. In Proceedings of the *International Workshop on Web Services Modeling and Testing (WS-MaTe)*, 2006, pp. 51-63.

[6] Z. Liu, T. Liu, L. Cai, and G. Yang. Quality Evaluation and Selection Framework of Service Composition Based on Distributed Agents. In Proceedings of the Fifth *International Conference on Next Generation Web Services Practices*, 2009, pp. 68-75.

[7] G. Canfora, M. Di Penta, R. Esposito, and M. L. Villani. An Approach for QoS-aware Service Composition based on Genetic Algorithms. In Proceedings of the *Genetic and Evolutionary Computation Conference (GECCO)*, 2005, pp. 1069-1075.

[8] M. Mitchell. *An Introduction to Genetic Algorithms.* The MIT Press, 1998.

[9] World Wide Web Consortium (W3C). http://www.w3.org/

[10] E. Cerami, and S. St. Laurent. Web services essentials, *O' Reilly and Associates, Inc.*, ISBN:0596002246, 2002.

[11] Faiz Gallouj. Innovation in the service economy: the new wealth of nations. *Edward Elgar Publishing*, 2002.

[12] S. Vandermerwe, and J. Rada. Servitization of Business: Adding Value by Adding Services. *European Management Journal*, Volume 6, No. 4, 1988.

[13] L. Zeng, B. Benatallah, M. Dumas, J. Kalagnanam, and Q. Z. Sheng. Quality driven web services composition. In Proceeding of the *International World Wide Web Conference*, 2003, pp.411-421.

[14] A. Alamari, M. Eid, A. E. Saddik. Classification of the state-of-the-art dynamic web services composition techniques. *International Journal of Web and Grid Services*, Vol. 2, No. 2, 2006, pp. 148-166.

[15] S. Ghandeharizadeh, C. A. Knoblock, C. Papadopoulos, C. Shahabi, E. Alwagait, J. L. Ambite, M. Cai, C. C. Chen, P. Pol, R. Schmidt, S. Song, S. Thakkar, and R. Zhou. Proteus: A system for dynamically composing and intelligently executing web services. In Proceedings of the first *International Conference on Web Services*, 2003, pp. 1-5.

[16] J. Bosch. Superimposition: A component adaption technique. *Journal of Information and Software Technology*, Vol. 41, 1999, pp. 257-273.

[17] F. Lecue. Optimizing QoS-aware semantic web service composition. In Proceedings of the *Eighth International Semantic Web Conference (ISWC)*, 2009.

[18] I. B. Arpinar, B. A. Meza, R. Zhang, and A. Maduko. Ontology driven web service composition platform. In Proceedings of the *IEEE International Conference on E-Commerce Technology*, 2004.

[19] A. Albreshne, and J. Pasquier. Semantic based semi-automatic web service composition. In Proceedings of the Fifth *Libyan Arab International Conference On Electrical and Electronic Engineering (LAICEEE)*, 2010, pp. 603-615.

[20] M. Mrissa, D. Benslimane, Z. Maamar, and C. Ghedira. Towards a semantic and context based approach for composing web services. *International Journal of Web and Grid Services*, Vol. 1, Nos. 3/4, 2005, pp. 268-286.

[21] D. VanderMeer, A. Datta, K. Dutta, H. Thomas, K. Ramamritham, S. B. Navathe. FUSION: A system allowing dynamic web service composition and automatic execution. Proceedings of the *International Conference on E-commerce*, 2003, pp. 399-404.

[22] N. Channa, S. Li, A. W. Shaikh, and X. Fu. Constraint satisfaction in dynamic web service composition. In Proceedings of the Sixteenth *International Workshop on Database and Expert Systems Applications*, 2005.

[23] B. Benatallah, M. Dumas, Q. Z. Sheng, and A. H. H. Ngu. Declarative composition and peer-to-peer provisioning of dynamic web services. In Proceedings of the 18th *International Conference on Data Engineering (ICDE)*, 2002.

[24] V. Chvatal. Linear Programming. W. H. Freeman, 1983.

[25] F. H. Khan, M. Y. Javed, S. Bashir, A. Khan, and M. S. H. Khiyal. QoS Based Dynamic Web Services Composition and Execution. *International Journal of Computer Science and Information Security*, Vol. 7, No. 2, 2010, pp.147-152.

[26] F. Rosenberg, P. Leitner, A. Michlmayr, P. Celikovic, and S. Dustdar. Towards Composition as a Service - A Quality of Service Driven Approach. In Proceedings of the Twenty-fifth *International Conference on Date Engineering*, 2009, pp.1733-1740.

[27] H. Chang, and K. Lee. A Quality-Driven Web Service Composition Methodology for Ubiquitous Services. *Journal of Information Science and Engineering*, Vol. 26, 2010, pp.1957-1971.

[28] M. Comuzzi, C. Kotsokalis, C. Rathfelder, W. Theilmann, U. Winkler, and G. Zacco. A Framework for Multi-level SLA Management. In Proceedings of the Eighth *International Conference on Service Oriented Computing*, 2010, pp.187-196.

[29] Reliability. In Wikipedia. Retrieved March 14, 2011, from http://en.wikipedia.org/wiki/Reliability

[30] P. Battilani, and F. Fauri. The rise of a service-based economy and its transformation: the case of rimini. *Ramini center for economic analysis, Working Paper Series*, 2007.

[31] J. Kilburn. The Decline of the Mills and the Rise of Social Services in a Northeastern Mill Town. *Annual meeting of the American Sociological Association*, 2003.

[32] M. P. Papazoglou. Service-Oriented Computing: Concepts, Characteristics and Directions. In Proceedings of the Fourth *International Conference on Web Information Systems Engineering*, 2003, pp. 3-12.

[33] S. Ran. A model for web services discovery with QoS. In Proceedings of the *ACM SIGecom Exchanges*, 2003, pp. 1-10.

[34] N. Kokash. Web service dicovery with implicit QoS filtering. In Proceedings of the *IBM PhD Student Symposium*, in conjunction with the *International Conference on Service Oriented Computing (ICSOC)*, 2005, pp. 61-66.

[35] F. Curbera, M. Duftler, R. Khalaf, W. Nagy, N. Mukhi, and S. Weerawarana. Unraveling the Web Services Web: An Introduction to SOAP, WSDL, and UDDI. In Proceedings of the *International Conference of Internet Computing*, 2002, pp. 86-93.

[36] http://aws.amazon.com/ec2/

[37] http://www.rightscale.com

[38] TRW Defense and Space Systems Group. Software Reliability Study. Redondo Beach, CA, 1976.

[39] M. Godse, R. Sonar, and S. Mulik. The Analytical Hierarchy Process Approach for Prioritizing Features in the Selection of Web Service. In Proceedings of the Sixth *European Conference on Web Services*, 2008, pp. 41-50.

[40] M. Fernndez, D. Vallet, and P. Castells. Probabilistic Score Normalization for Rank Aggregation. In Proceedings of the 28th *European Conference on Information Retrieval*, 2006, pp. 553-556.

[41] A. R. Callaghan, and K. E. Lewis. A 2-Phase aspiration-level and utility theory approach to large scale design. In Proceedings of the *Design Engineering Technical Conferences And Computers and Information in Engineering Conference*, 2000.

[42] M. Kulok, and K. E. Lewis. Preference consistency in multiattribute decision making. In Proceedings of *Design Engineering Technical Conferences And Computers and Information in Engineering Conference*, 2005.

[43] D. Luenberger. Linear and Nonlinear Programming. Addison-Wesley, Reading (MA), 1984.

[44] C. L. Hwang and J. L. Williams and L. T. Fan. Introduction to the generalized reduced gradient method. Institute for Systems Design and Optimization, Kansas State University (Manhattan), 1972

[45] Y. Liu, A. H.H. Ngu, and L. Zeng. QoS Computation and Policing in Dynamic Web Service Selection. In Proceedings of the *International World Wide Web conference*, 2006, pp. 390-401.

[46] X. Wang, T. Vitvar, M. Kerrigan, and I. Toma. A QoS-aware Selection Model for Semantic Web Services. In Proceedings of the *International Conference on Service-Oriented Computing*, 2004.

[47] E. Al-Masri, and Q. H. Mahmoud. Discovering the Best Web Service. In Proceedings of the *International World Wide Web Conference*, 2007.

[48] F. A. Salustri. Pairwise Comparison. http://deed.ryerson.ca/ fil/t/pwisecomp.html

[49] A. Ankolekar, M. Burstein, J. R. Hobbs, O. Lassila, D. McDermott, D. Martin, S. A. McIlraith, S. Narayanan, M. Paolucci, T. Payne, and K. Sycara. DAML-S: Web service description for the semantic web. In Proceedings of the *International Semantic Web Conference (ISWC)*, 2002, pp. 348-363.

[50] G. G. Yin, Q. Zhang. Continuous-time markov chains and applications: A singular perturbation approach. *Stochastic Modeling and Applied Probability Series*, Vol. 37, ISBN: 978-0-387-98244-1, Springer, 1998.

[51] G. Balbo. Introducation to stochastic petri nets. Lectures on *Formal Methods and Performance Analysis: First EEF/Euro Summer School on Trends in Computer Science*, 2002, pp. 84-155.

[52] D. Zhovtobryukh. A petri net-based approach for automated goal-driven web service composition. *International Society for Computer Simulation*, Volume 83, Issue 1, 2007, pp.33-63.

[53] K. H. Kim. Toward QoS certification of real-time distributed computing systems. In Proceedings of the 7th IEEE *International Symposium on High Assurance Systems Engineering (HASE)*, 2002.

[54] G. G. Yin, and Q. Zhang. Discrete-time markov chains: Two-time-scale methods and applications. *Stochastic Modeling and Applied Probability Series*, Vol. 55, ISBN: 978-0-387-21948-6, Springer, 2005.

[55] W. H. Greub. Linear Algebra. Springer, 1981.

[56] W. J. Stewart. Introduction to the numerical solution of markov chains. CRC press, 1991.

[57] M. Sniedovich. Dynamic Programming. CRC press, 1992.

[58] T. Yu, and K-J. Lin. Service selection algorithms for Web services with end-to-end QoS constraints. In Proceedings of the *International Conference on e-Commerce Technology*, 2004.

[59] W.T. Tsai, D. Zhang, Y. Chen, H. Huang, R. Paul, and N. Liao. A Software Reliability Model for Web Services. In Proceedings on *Software Engineering and Applications (SEA)*, 2004.

[60] W-L. Wang, Y. Wu, and M-H. Chen. An Architecture-Based Software Reliability Model. In Proceedings of the 12th *International Conference on World Wide Web*, 2003.

[61] Z. Gao, and G. Wu. Combining Qos-based service selection with performance prediction. In Proceedings of the IEEE *International Conference on e-Business Engineering (ICEBE)*, 2005, pp. 611-614.

[62] I. Epifani, C. Ghezzi, and R. Mirandola. Model Evolution by Run-Time Parameter Adaptation. In Proceedings of the *International Conference on Software Engineering*, 2009.

[63] C. Wu, Service distribution and service discovery through a public web services platform. Ph.D. Thesis, Curtin University of Technology, Digital Ecosystems and Business Intelligence Institute, 2008.

[64] A. Ismail, J. Yan, and J. Shen. Dynamic Service Selection for Service Composition with Time Constraints. In Proceedings of the *Australian Software Engineering Conference*, 2009.

[65] X. Wang, K. Yue, J. Z. Huang, and A. Zhou. Service Selection in Dynamic Demand-Driven Web Services. In Proceedings of the *International Conference on Web Services*, 2004.

[66] Z. S. Bahjat, and G. B. Fried. Automatic selection of different motion profile parameters based on average waiting time. United States Patent 5290976.

[67] D. Gross, and C. M. Harris. Fundamentals of Queueing Theory. Wiley Series in Probability and Statistics, 1998.

[68] R. J. Wonnacott, and T. J. Wonnacott. Introductory Statistics. John Wiley and Sons, 1985.

[69] X. Gao, Y. Asami, and C. J. Chung. An empirical evaluation of hedonic regression models. *Symposium on Geospatial Theory*, 2002.

[70] L. Green. Queueing Analysis in Healthcare. *International Series in Operations Research and Management Science*, 2006.

[71] A. D. Flockhart, R. H. Foster, R. A. Jensen, J. E. Kohler, and E. P. Mathews. Call center agent selection that optimizes call wait times. United States Patent 6192122.

[72] http://www.castiron.com

[73] X. Chen, and P. G. Sorenson. Survey of organizations using SaaS. To be published.

[74] C. Z. Mooney, and R. D. Duval. Bootstrapping: A nonparametric approach to statistical inference. *Quantitative applications in social sciences series*, Sage publications, 1993.

[75] B. K. Youse. Mathematical Induction. Prentice-Hall Inc., Englewood Cliffs, N. J., 1964.

[76] J. H. Smith. A treatise on arithmetic. Rivington's Mathematical Series, 1873.

[77] T. H. Cormen. Introduction to algorithms. The MIT press, 1990.

[78] http://www.service-level-agreement.net/

[79] "Reputation." Wikipedia, 2010. Wikimedia Foundation Inc., 18th May, 2010. http://en.wikipedia.org/wiki/Reputation

[80] Z. Malik, and A. Bouguettaya. RATEWeb: Reputation Assessment for Trust Establishment among Web services. The *VLDB Journal*, Volume 18, pp. 885911, 2009.

[81] S. Buchegger, and J-Y Le Boudec. A Robust Reputation System for P2P and Mobile Ad-hoc Networks. Workshop on the *Economics of Peer-to-Peer Systems (P2PEcon)*, 2004.

[82] S. Figini, P. Giudici, J. T. S. Gomez. Statistical Methods for Reputation Measurements. www-3.univpv.it/dipstea/workingpapers/

[83] S. Morinaga, K. Yamanishi, K. Tateishi, and T. Fukushim. Mining Product Reputations on the Web. *ACM Special Interest Group in Knowledge Discovery and Data Mining (SIGKDD)*, 2002.

[84] E. Chang, F. K. Hussain, and T. Dillon. Reputation Ontology for Reputation Systems. *On the Move to Meaningful Internet Systems: OTM Workshops*, 2005.

[85] J. M. Pujol, R. Sanguesa, and J. Delgado. Extracting Reputation in Multi Agent Systems by Means of Social Network Topology. Proceedings of the *International Conference on Autonomous Agents and Multiagent Systems*, 2002.

[86] http://www.tripadvisor.com/

[87] http://office.microsoft.com/en-us/excel/

[88] http://www.btnonline.com/

[89] W. S. Cleveland. Robust locally weighted regression and smoothing scatterplots. *Journal of American Statistical Association*, pp. 829-836, Volume 74, Number 368, 1979.

[90] V. Deora, J. Shao, G. Shercliff, P.J. Stockreisser, W.A. Gray and N.J. Fiddian. Incorporating QoS specifications in service discovery. *Web Information Systems (WISE)*, 2004, pp. 252-263.

[91] V. Kumar. Algorithms for Constraint Satisfaction Problems: A survey. A. I. Magazine, 1992, pp. 32-44.

[92] www.expedia.com

[93] www.eurail.com

[94] H. A. Taha. Operations Research: An Introduction. 8th Edition, Prentice-Hall, 2006.

[95] O. J. Blanchard, and D. Quah. The Dynamic Effects of Aggregate Demand and Supply Disturbances. *The American Economic Review*, Volume 75, Number 4, pp. 655-673, 1989.

[96] R. G. Edmonds. A Theoretical Basis for Hedonic Regression: A Research Primer. *Real Estate Economics*, Volume 12, Issue 1, pp. 72-85, 1984.

[97] http://maps.google.com/

[98] A. Srivastava and P. G. Sorenson. The Influence of Service Interactions on Individual Service Reliability in a Composition Scenario. In Proceedings of the 6th *International Conference on Web Information Systems and Technologies (WEBIST)*, 2010, pp. 30-39.

[99] A. Srivastava and P. G. Sorenson. Utilizing the Waiting-time Criterion for Selecting Services in a Composition Scenario. In Proceedings of the 7th *International Conference on Services Computing (SCC)*, Application and Industry Track, 2010, pp. 258-264.

[100] A. Srivastava and P. G. Sorenson. Utilizing Multiple Regression to Assess Service Reputation. International Conference on Predictive Models in Software Engineering (Submitted)

[101] A. Srivastava and P. G. Sorenson. The Validation of a Dynamic Service Composition Model using a Simple Game. In Proceedings of the *International Conference on Knowledge Management and Information Sharing (KMIS)*, 2010, pp. 260-266.

[102] A. Srivastava and P. G. Sorenson. Service Reliability Assessment in a Composition Environment. *Lecture Notes in Business Information Processing*, Springer-Verlag, Volume 75, 2011, pp. 30-45.

[103] A. Srivastava and P. G. Sorenson. Service Selection based on Customer Rating of Quality of Service Attributes. In Proceedings of the 8th *International Conference on Web Services (ICWS)*, 2010, pp. 1-8.

Notification of Ethics Delegated Approval

Study ID:	Pro00017424
Study Title:	Dynamic Service Composition Simulation Study
Study Investigator:	Abhishek Srivastava
Supervisor:	Paul Sorenson
Funding/Sponsor:	NSERC - Natural Sciences And Engineering Research Council NSERC
Approval Expiry Date:	November 29, 2011

Thank you for submitting the application above to the Arts, Science, Law REB. Theresa Garvin has reviewed your application for human research ethics and finds that your proposed research meets the University of Alberta standards for research involving human participants (GFC Policy Section 66). On behalf of the Arts, Science, Law REB, I am providing **delegated research ethics approval** for your proposed research.

The research ethics approval is valid for one year and will expire on November 29, 2011.

A request for renewal must be submitted prior to the expiry of this approval if your study still requires ethics approval at that time. If you do not renew before the renewal expiry date, you will have to re-submit an ethics application.

If there are changes to the project that need to be reviewed, please file an amendment. If any adverse effects to human participants are encountered in your research, please contact the undersigned immediately.

Sincerely,

Dr. Nancy Lovell
Chair, Arts, Science, Law REB

Note: This correspondence includes an electronic signature (validation and approval via an online system).