Chapter 4
Horizontal Service Composition for Language Services

Ahlem Ben Hassine¹, Shigeo Matsubara², and Toru Ishida²

¹ National School of Computer Science (ENS1), Tunis University, Tunisia, e-mail: ahlembh@gmail.com
² Department of Social Informatics, Kyoto University, Yoshida-honmachi, Sakyo-ku, Kyoto 606-8501, Japan, e-mail: {matsuba, ishida}@i.kyoto-u.ac.jp

Abstract In the Language Grid, automatically composing Web services is a crucial task. This task involves vertical and horizontal composition. Vertical composition consists of defining an appropriate combination of simple processes to perform a composition task. Horizontal composition consists of determining the most appropriate Web service from among a set of functionally equivalent ones for each component process. The latter is important in language services. For the horizontal composition of Web services, we propose a generic formalization of any Web service composition problem based on a constraint optimization problem (COP) and then propose an incremental user-intervention-based protocol to find the optimal composite Web service according to some predefined criteria at run-time.

4.1 Introduction

The Language Grid provides users with functions to combine language resources (e.g., bilingual dictionaries) or language processing functions (e.g., machine translators) and to add their own language resources to create new language services for their own intercultural activities (Ishida 2006). That is, combining a variety of language services allows users to make better use of the large quantity of language resources that have accumulated on the Internet. It will enable a language service to be built that is optimal for the actual field of activity performed by the intercultural collaboration.

Consider a specialized translation service with back translations. This service can be achieved by using a composite service. Several atomic services, such as machine translations, morphological analyzers, and specialized dictionaries can be combined to create the specialized translation service. However, this composition task might be difficult to realize because so many services with the same or similar ability exist. To overcome this difficulty and properly support users, we have developed a constraint-based Web service composition technique.
Our technique is based on the technologies of Web services. The great success of Web services, due especially to their rich applications made possible by open common standards, has led to their wide proliferation and a tremendous variety of Web services are now available. However, this proliferation has rendered the discovery and use of the most appropriate Web service arduous. These tasks are increasingly complicated, especially if the target is a composite Web service that must satisfy a user’s long-term complex goal. The automatic Web service composition task consists of finding an appropriate combination of existing Web services to achieve a global goal.

We focus on the fact that many available Web services can fulfill the same task and we refer to these Web services as functionally equivalent Web services. In the sequel of this chapter, as is generally done in the literature, we refer to each of the subtasks making up the main goal as an abstract Web service and to each Web service able to perform a subtask as a concrete Web service. Solving the Web service composition problem involves two types of composition:

- **Vertical** composition is aimed at finding the “best” combination of abstract Web services, i.e., abstract workflow, in terms of achieving the main goal while satisfying all existing interdependent restrictions.
- **Horizontal** composition is aimed at finding the “best” concrete Web service, from among a set of available functionally equivalent Web services, i.e., executable workflow, to perform each abstract Web service. The quality of the response to the user’s query (the composition task) strongly depends on the selected concrete Web services. The choice of a concrete Web service is dictated by functional (i.e., related to the inputs) and/or non-functional attributes (i.e., related to the quality of service attributes).

The main benefits gained by differentiating these two composition processes are: i) the Web service composition problem is simplified with reduced computational complexity, ii) avoiding any horizontal composition redundancy that may appear while searching for the “best” combination of abstract Web services, and mainly iii) ensuring more flexibility for user intervention, i.e., the user is able to modify/adjust the abstract workflow when needed.

This chapter consists of two main parts. The first is a generic formalization of any Web service composition problem as a constraint optimization problem (COP) in which we try to express most of the Web service composition problem features in a simple and natural way. Our main purpose is to develop a common and robust means of expressing any Web service composition problem that ideally reflects realistic domains. The second contribution is a real-time interactive protocol to solve any Web service composition problem by overcoming most of the limitations encountered above. Although there are various techniques for solving COPs, none of these consider the user interaction issue. The constraint optimization problem formalism is especially promising for ideally describing any realistic Web service composition problem, because this problem is a combinatorial problem that can be represented by a set of variables connected by constraints. Two approaches are proposed in this chapter, a centralized approach and a distributed approach.