Computer Communications 34 (2011) 159–168

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

Towards runtime discovery, selection and composition of semantic services

Eduardo Gonçalves da Silva *, Luís Ferreira Pires, Marten van Sinderen

Centre for Telematics and Information Technology, University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands

ARTICLE INFO

Article history:
Available online 7 April 2010

Keywords:
Service-oriented computing
User-centric service composition
Semantic services

ABSTRACT

Service-orientation is gaining momentum in distributed software applications, mainly because it facilitates interoperability and allows application designers to abstract from underlying implementation technologies. Service composition has been acknowledged as a promising approach to create composite services that are capable of supporting service user needs, possibly by personalising the service delivery through the use of context information or user preferences. In this paper we discuss the challenges of automatic service composition, and present DynamiCoS, which is a novel framework that aims at supporting service composition on-demand and at runtime for the benefit of service end-users. We define the DynamiCoS framework based on a service composition life-cycle. Framework mechanisms are introduced to tackle each of the phases and requirements of this life-cycle. Semantic services are used in our framework to enable reasoning on the service requests issued by end-users, making it possible to automate service discovery, selection and composition. We validate our framework with a prototype that we have built in order to experiment with the mechanisms we have designed. The prototype was evaluated in a testing environment using some use case scenarios. The results of our evaluation give evidences of the feasibility of our approach to support runtime service composition. We also show the benefits of semantic-based frameworks for service composition, particularly for end-users who will be able to have more control on the service composition process.

1. Introduction

With the Internet becoming ubiquitous, the use of network-based application services is being increasingly adopted and it is expected to grow in the upcoming years [1]. This is being reflected in many technology developments and innovations, such as, for example, *Software as a Service* (SaaS) [2], *Internet of Services* [3] and *Cloud Computing* [4]. The proliferation of service-oriented systems [5] is leading to the emergence of large sets of services in different domains. At the same time, the use of mobile devices with fast data connections is increasing quite rapidly. In [6] it is reported that by 2013 more than 38% of the European population will access the Internet on their mobile device, which is an increase of 300% compared to the current situation.

These developments are allowing and *pushing* new, more adaptive and personalised application services where the users play an active role in the process of service creation. This is one of the main motivations behind the *Internet of Services*. However, users are not expected to create new services from scratch but to *aggregate* existing services to fulfil a set of user requirements. Supporting end-users in this kind of runtime service creation process is a complex undertaking. Different users have different preferences and request services in different context situations, which require different actions to be taken. Furthermore, end-users expect a high-level of abstraction in the service creation process, since they lack the technical knowledge to use advanced technical tools. This implies that automation has to be provided to support the end-user in the service creation process. We claim that this can be achieved by using semantic-based service composition approaches. We assume that if no single application service exists to provide a requested service to a user, a new composite service can possibly be created on-demand from existing services, considering the user preferences and context to personalise the service creation process. If a user request cannot be fully matched in the creation process, a partial fulfilment of the user request may be possible or an alternative suggestion that the user is not aware of may be proposed. We denote the creation of service compositions on-demand based on specific requirements as *dynamic service composition*. To support this approach, we developed a framework for dynamic service composition provisioning called *DynamiCoS*.

The *DynamiCoS* (*Dyna*mic *Com*position of *S*ervices) framework addresses all the phases and stakeholders of the dynamic service composition life-cycle. To allow automation of the composition life-cycle, semantic information is used based on ontologies (domain conceptualisations) to which the different framework stakeholders have to comply. The framework allows service developers

* Corresponding author.
E-mail addresses: e.m.g.silva@ewi.utwente.nl (E.G. da Silva), l.ferreirapies@ewi.utwente.nl (L.F. Pires), m.j.vansinderen@ewi.utwente.nl (M. van Sinderen).

0140-3664/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.comcom.2010.04.003

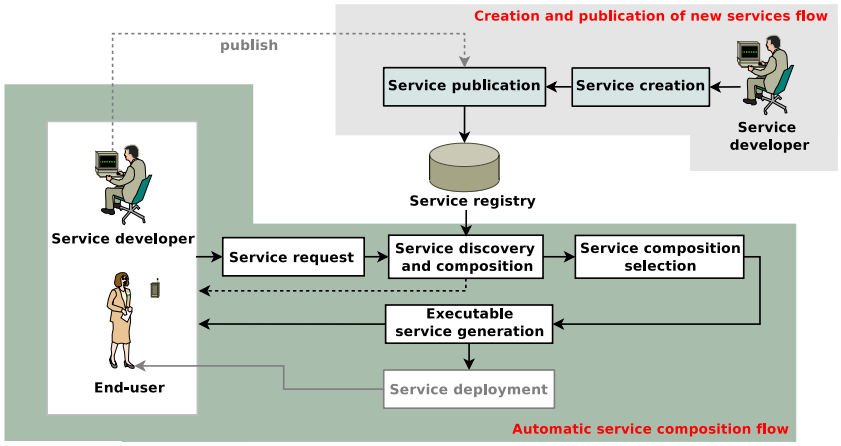

Fig. 1. Dynamic service composition life-cycle.

to publish their semantically annotated services in a formalism that is neutral with respect to description languages and technologies, which consequently enables the use of different semantic service description languages. The composition process is likewise language-neutral, so that services described in different service description languages can be combined in the same service composition. DynamiCoS supports end-users in the service creation process, through automatic discovery, selection and composition of services based on the user service request. We make use of the notion of *goal* to describe and specify the activities (or operations) the services can perform and to capture the requirements the user wants to fulfil when executing a service. We argue that comprehensive frameworks, which address not only composition but also the other supporting phases of the life-cycle, are required to boost the use of automated mechanisms to support service composition, especially at runtime.

This paper is further organised as follows: Section 2 presents our dynamic service composition life-cycle; Section 3 presents our approach to address the dynamic service composition life-cycle, and the details of the DynamiCoS conceptual framework and prototype; Section 4 presents the testing environment and the methodology used to evaluate our approach; Section 5 presents and analyses the evaluation results; Section 6 provides an overview of related work; and Section 7 gives our conclusions and challenges for future work.

2. Dynamic service composition life-cycle

Fig. 1 presents our dynamic service composition life-cycle, depicting the phases and stakeholders associated with the service composition process [7].

We consider two stakeholders in this life-cycle: *Service developer*, who publishes new services in the framework, which can be used as basic components on the composition process, or creates new service compositions at design-time using the framework support; and *End-user*, who makes use of automatic service composition mechanisms at runtime. The life-cycle has two main flows, namely *creation and publication of new services* and *automatic service composition*.

The *service creation* phase is performed by a service developer, who creates new services by programming new applications and makes them available as services, or builds a new service composition from existing services and makes the resulting composition available as a new service. The service creation phase also encompasses the definition of the service description document by the service developer. The service description document is then used in the *service publication* phase to publish the service functional and non-functional information in a service registry. The definition of a service description document, and specifically a semantic service description, is compulsory to enable automation in the service composition process. If services are not semantically described, then automation cannot be achieved in the service composition process.

In Fig. 1, the *automatic service composition* flow assumes that an end-user or a service developer wants a new service to satisfy some specific requirements. We assume that an end-user has no technical skills on service composition and wishes a new service at *runtime*. A service developer is a user with technical skills on service composition, but wants to create a new service at *design-time*, in a faster (and more automated) way, given some specific requirements for a new service. The first phase of the automatic service composition process is the specification of a *service request*, where the user specifies requirements and preferences for the desired service. Once the service request is defined, the *service discovery and composition* phase takes place. Candidate services are discovered through the service registry interface according to some properties defined by the user. Services that match these properties are discovered from the service registry. Given the set of discovered services, an algorithm takes the user service request into account and builds candidate compositions. Interactions may take place with the user to guide and refine the composition process in case the algorithm cannot deliver compositions that fulfil the service request. Once more than one service compositions are found that match the user service request, the *service composition selection* phase takes place. In the case of an end-user, a single service (composition) is expected to be returned, i.e., the system has to select one service, possibly based on the user service request, preferences and context. In the case of a service developer, a ranked list of services that match the service request may be returned. The *Executable service generation* phase consists of the creation of an executable representation of the generated service compositions. This is necessary because usually the core composition process is performed using a formal representation of the service compositions that is not executable. In the *Service deployment* phase the selected composition is deployed to allow the execution and delivery of the created service.

3. DynamiCoS framework

DynamiCoS[1] (*Dynamic Co*mposition of *S*ervices) is a framework for the provision of dynamic service composition that supports all the life-cycle phases and stakeholders identified in Section 2. Fig. 2

[1] http://dynamicos.sourceforge.net