Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università degli Studi di Roma "La Sapienza"

Negotiation among Web Services using LOTOS/CADP

Gwen Salaün, Andrea Ferrara, Antonella Chirichiello

Technical Report n. 13
2004

Aracne

I *Technical Reports* del Dipartimento di Informatica e Sistemistica "Antonio Ruberti" svolgono la funzione di divulgare tempestivamente, in forma definitiva o provvisoria, i risultati di ricerche scientifiche originali. Per ciascuna pubblicazione vengono soddisfatti gli obblighi previsti dall'art. 1 del D.L.L. 31.8.1945, n. 660 e successive modifiche.
Copie della presente pubblicazione possono essere richieste alla Redazione.

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università degli Studi di Roma "La Sapienza"

Via Eudossiana, 18 - 00184 Roma
Via Buonarroti, 12 - 00185 Roma
Via Salaria, 113 - 00198 Roma

www.dis.uniroma1.it

www.aracne–editrice.it
info@aracne–editrice.it

Redazione:
00173 Roma
via Raffaele Garofalo, 133 A/B
(06) 72672222 – (06) 93781065
telefax 72672233

ISBN 88–7999–755-6

I edizione: maggio 2004

Finito di stampare nel mese di giugno del 2004
dalla tipografia « Grafica Editrice Romana S.r.l. » di Roma
per conto della « Aracne editrice S.r.l. » di Roma
Printed in Italy

Negotiation among Web Services using LOTOS/CADP

Gwen Salaün, Andrea Ferrara, Antonella Chirichiello

DIS - Università di Roma "La Sapienza"
Via Salaria 113, 00198 Roma, Italia
Emails: {salaun,ferrara}@dis.uniroma1.it,
anto.chirichiello@libero.it

Abstract

It is now well-admitted that formal methods are helpful for many issues raised in the web service area. In a previous work, we advocated the use of process algebra to describe, compose and reason on web services at an abstract level. In this paper, we extend this initial proposal, which only dealt with behavioural aspects, to cope with the question of representing data aspects as well. In this context, we show how the expressive process algebra LOTOS (and its toolbox CADP) can be used to tackle this issue. We illustrate the usefulness of our proposal on an important application in e-business: negotiation among web services. The connection between abstract specifications and running web services is made concrete thanks to systematic guidelines enabling one to map LOTOS and the executable language BPEL in both directions.

Keywords: Web Services, Formal Methods, LOTOS, CADP, Negotiation, BPEL.

1 Introduction

Web services (WSs) are distributed and independent pieces of code solving specific tasks which communicate with each other through the exchange of messages. A more unusual specificity that distinguishes them from more traditional software components is that they are deployed and then accessed through the internet. Some XML-based standardized technologies have already been proposed to support WSs development: WSDL interfaces abstractly describe messages to be exchanged, SOAP is a protocol for exchanging structured information, UDDI is a repository to publish and discover WSs, BPEL4WS (BPEL for short) is a notation for describing executable business process behaviours. WSs raise many theoretical and practical issues which are part of on-going research. Some well-known problems related to WSs are to specify them in an adequate, formally defined and expressive enough language, to compose them (automatically), to discover them through the web, to ensure their correctness, etc.

Formal methods provide an adequate framework (many specification languages and reasoning tools) to address most of these issues (description, composition, correctness). Different proposals have emerged recently to abstractly describe WSs, most of which are grounded on transition system models (Labelled Transition Systems, Mealy automata, Petri nets, etc.) [2, 17, 25, 15, 20] and to verify WS description to ensure some properties on them [25, 11, 24]. In a previous work [26], we advocated the use of process algebra (PA) [3] for WSs. Being simple, abstract and formally defined, process algebras make it easier to formally specify the message exchange between WSs, and to reason on the specified systems (*e.g.* using bisimulation notions to ensure the correctness of composition). PA also seems a suitable choice to the choreography issue, that need to describe public interfaces (observable behaviours) of WSs.

In the initial proposal [26], we especially experimented the use of the simple process algebra CCS. However, CCS turns out to be only adequate for the specification of (and reasoning on) dynamic behaviours. What was missing in this proposal was to handle *data*. This allows a much finer (less abstract) level of specification, which is clearly needed in some cases. In this paper, we argue that the process algebra LOTOS [18, 5] and its toolbox CADP [13] are useful respectively to describe WSs and to reason on them.

We also propose a two-level description of WSs: an abstract one (using LOTOS) and an executable one (using WSDL and BPEL)[1]. Following such an approach, we can develop WSs considering the formal and verified specification as a starting point. In the other direction, we can abstract a deployed system to a description in LOTOS. The interest of such an abstract description is that the formality of this language and its readily existing tools enable one to validate and verify specifications through animation and proofs of temporal properties.

To illustrate the interest of such an approach in WSs, we focus on the problem of *negotiation* in which both data and dynamic aspects have to be dealt with. The perspective of intelligent/automated WSs which would be able to automatically perform the necessary negotiation steps to satisfy their user's request in the most satisfactory possible way has been emerging from artificial intelligence and multi-agent systems. This problem is a typical example of services involving both data (prices, goods, etc) and behaviours. Negotiation issues appear when several participants (clients and providers) have to interact to reach an agreement that is beneficial to all of them. The classical case is the sale/purchase framework. Note that the most widespread schema of interaction is the peer to peer interaction between one requester and one provider. However, negotiation arises in many contexts and with many variants; we will evoke them further in this paper. Our goal is to show how LOTOS/CADP may be used to ensure trustworthy negotiation steps. Such a validation stage is mandatory to ensure the running of negotiating WSs in an automated and correct way.

The organization of this paper is as follows. First, we introduce in Section 2 the different entities involved in negotiation. Section 3 presents the LOTOS language and its associated toolbox CADP. They are used in Section 4 to describe our negotiating processes at an abstract level, and to reason on them. Section 5 gives some systematic guidelines to translate LOTOS to BPEL or to abstract LOTOS specifications from BPEL code. Related works are in-

[1] In the remainder of the paper, terms *processes* and *WSs* are used as synonyms even if *processes* is maybe more adequate for LOTOS abstract descriptions and *WSs* for BPEL executable services.

troduced in Section 6 and compared with the current proposal. Finally, we draw up some concluding remarks in Section 7.

2 What does Negotiation Involve?

In this section, we introduce what is needed to describing negotiation cases. This short overview draws its inspiration from [7]. These aspects will next be specified using LOTOS (Section 4) and encoded in BPEL (Section 5).

Variables. They represent entities on which processes should negotiate. A common example is the price of something which could be sold on-line. However, many variables may be involved in a negotiation representing for instance the availability of different products at the same time, the fees of delivery, or the maximum number of days for a delivery.

Constraints. They represent conditions to respect (in which case they are called invariants) while trying to reach an agreement. Such an invariant is actually a predicate which can be evaluated replacing free variables with actual values. We can imagine some quite complex invariant involving values of different variables. For a requester who is trying to buy by auction a product, such an invariant could be that (s)he is ready to pay 300 euros at most with a delivery within 10 days, or to accept a delivery higher than 10 days if there is a price reduction of 10% at least.

Exchanged information. In order to reach an agreement, both participants should send values to each other. A simple case could be a price, but they can also exchange more advanced constructs (a record of values, a constraint on a value, etc).

Strategies. As written in [22]: *"An agent's negotiation strategy is the specification of the sequence of actions (usually offers or responses) the agent plans to make during the negotiation."*. Strategies may take into account other considerations. For instance, a participant can try to reach an agreement as soon as possible, or to minimize a price. Therefore, strategies are related to minimizing or maximizing objective functions as well.