Evaluation of Large Scale Web Service Compositons

Thiago Rodrigues Colucci

Advisor: Fabio Kon

September 10, 2010

Contents

Alumnus: Thiago Rodrigues Colucci
Advisor: Prof. Dr. Fabio Kon
THIS DOCUMENT IS IN DEVELOPMENT, THE SECTIONS LEFT IN PORTUGUESE SHOULD BE IGNORED

1 Introduction

With the development of the Internet, we've been able to use a lot of Web Applications (as "Gmail", "Google Docs" and "Amazon.com"). Those applications use a lot of web services for decoupling it's parts. Now that we have plenty of services to use, we need some technic to compose those web services into new ones with higher level of abstraction. In this context arose the Orchestration and the Choreography of Web services, technics to create services using only description languages and pre-existing web services.

Business process built using compositions become interoperable and scalable, by the fact that they depend basically upon web services. Besides, there's already a lot of open web services available to use, shortening the development time.

With the grown importance of those compositions technics, this work will study a little of the differences between them and also evaluate large scale synthetical compositions. This evaluation assist the development of the project *Baile*, Institute of Mathematics and Statistics from University of São Paulo (IME-USP) in partnership with Hewlett-Packard (HP) of the United States, as well as the project *CHOReOS*, IME-USP in partnership with a European consortium.

Baile's main goal is to study the problems related to the development of choreography in large scale environments, particularly in the context of Cloud Computing. The ultimate goal of CHOReOS, for USP, will be to implement a Service Oriented Middleware for the "Internet of the Future", an *Ultra-Large Scale internet*, heterogeneous and dynamic.

Also, by creating those compositions, this work will help in studying the problems for Baile and show how scalable the current techniques of choreography and orchestration are, later helping CHOReOS to understand the current state of the art of these techniques of composition of web services.

2 Concepts and Used Technologies

In this section we'll describe and explain the main technologies used in this work.

2.1 Web Service

Web services are application components, that communicate using open protocols using XML as the basis for the message exchange. The great thing about these components is that they can be used by other applications with total decoupling.

Those characteristics of web services allow them to be interoperable, reusable and also connect existing software. **[WST]**

Interoperability When all major platforms could access the Web using Web browsers, different platforms could interact. For these platforms to work together, Web-applications were developed. Web-applications are simple applications that run on the web. These are built around the Web browser standards and can be used by any browser on any platform.

Reusable application-components There are things applications need very often. So why make these over and over again? Web services can offer application-components like: currency conversion, weather reports, or even language translation as services.

Connect existing software Web services can help to solve the interoperability problem by giving different applications a way to link their data. With Web services we can exchange data between different applications and different platforms.

The basic Web services platform is XML with HTTP. XML provides a language which can be used between different platforms and programming languages and still express complex messages and functions. The HTTP protocol is the most used Internet protocol, this way we don't get bothered with small things like firewall.

2.2 Orchestration

Orchestration describes how web services can interact with each other at the message level, including the business logic and execution order of the interactions. These interactions may span applications and/or organizations, and result in a long-lived, transactional, multi-step process model.

Orchestration is not defined as a W3C standard. It's standardized by OASIS. The co-chair of the Web Service Choreography working group of W3C (in 2005), Steve Ross-Talbot, said that "OASIS is a very different kind of organization. By contrast, it's much more vendor-led than the W3C. The W3C is responsible for all the core building blocks of the Web, for Web services and for the semantic Web. All the ancillary things you need to make Web services happen, such as management and orchestration, are in OASIS". **[INTERVIEW]**

But the fact is that the industry is embracing orchestration as the way to specify, generate and control business process built upon web services. This was an initiative of IBM with Microsoft to develop a unified standard for Orchestration.

Using open standards to connect web services together. Basically, we first describes the orchestration through a XML document, then we generate an executable process from this file, and finally we execute the process. A good analogy would be a *C* program, with only the main function (orchestration definition) that uses a lot of functions (operations) of included libraries (partner web services). The hole logic is embedded in this one description, a center node.

Historically (2002), three standards arose from the cooperation between companies (e.g. IBM, Microsoft, Sun, BEA, and others) interested on the development of the Web:

BPEL4WS stands for "Business Process Execution Language for Web Services", the specification — called BPEL, for short — models the behavior of Web services in a business process interaction. It provides an XML-based grammar for describing the control logic required to coordinate Web services participating in a process flow. The WSDL interface defines the specific operations allowed, and BPEL defines how to sequence them.

WSCI is the "Web Service Choreography Interface". It defines a collaboration extension to WSDL, describing only the observable behavior between Web services. It does not address the definition of executables business process as BPEL does. We'll describe choreographies later.

BPML the "Business Process Management Language" also incorporate WSCI support. BPML and WSCI share the same underlying process execution model, so developers can use WSCI to describe public interactions among business process and reserve BPML for developing private implementations.

In the beginning of 2003, all the companies supporting BPML discontinued it's support, migrating to BPEL4WS. It became a standard by OASIS, whom renamed the standard to WS-BPEL, the change was made to align with others Web Services standards.

Nowadays the main standard is still BPEL, with a lot of tools to write it's definition and deploy to executable process.

WS-BPEL Before telling how to use BPEL, we first have to define some therms:

basic activity is an instruction that interacts with something external to the process itself. In a typical scenario, a BPEL executable process receives a message. Then it might invoke a series of services to gather additional data and subsequently respond to the requestor.

structured activities manage the overall process flow, specifying the sequence for referenced Web services. These activities also support conditional looping and dynamic branching. They are essentially BPEL's underlying programming logic for BPEL.

variables identifies the specific data exchanged in a message flow. When a BPEL process receives a message, it populates the appropriate variable so that subsequent requests can access the data. Variables are used to manage data persistence across Web services requests.

partnerLink can be any service that the process invokes or any service that invokes the process. Each partnerLink maps to a specific role that it fulfills in the business process.

scope tag To group a set of activities in a single transaction, BPEL uses a *scope* tag. The tag signifies that the enclosed steps should either all complete or all fail. Within this scope, a developer can specify compensation handlers that the BPEL orchestration engine can invoke in case of error.

Now, with our commands well known, we can analyze an BPEL description example.

Explicação do que são serviços web, como os implementamos, definições de orquestração, coreografias, computação nas nuvens e a "Internet do Futuro". Com isto o leitor compreenderá as demais seções sem dificuldades.

3 Atividades realizadas

Descrição de como foram criadas as composições de serviços web e experimentos realizados no processo inteiro.

4 Resultados e produtos obtidos

Nesta seção estarão, basicamente, os resultados dos testes.

5 Conclusões

6 Bibliography

WST Web Service Tutorial - `http://www.w3schools.com/webservices/`

WSO C. Peltz. Web services orchestration: a review of emerging technologies, tools, and standards. Technical Report HP. January, 2003

WSOC C. Peltz. Web Services Orchestration and Choreography. IEEE Computer Society, 20c.03.

NIMSA Non-intrusive monitoring and service adaptation for WS-BPEL. 17th international conference on World Wide Web, 2008

IWSA Gottschalk et al. Introduction to Web service architecture. IBM Systems Journal, Vol 41, NO 2, 2002.

WSBCBC M. Stal. Web Services: Beyond Component-based Computing. Communications of the ACM, Vol 45, NO 10, 2002.

OASIS OASIS(Organization for the Advancement of Structured Information Standards) - `http://www.oasis-o`

BPEL BPEL - `bpel.xml.org`

W3C W3C - `www.w3.org/TR`

INTERVIEW Interview with Steve Ross-Talbot - `http://searchsoa.techtarget.com/news/interview/0,289202,s`

BPSHO Business Process Spec Handed Off to OASIS, Not W3C - `http://www.internetnews.com/dev-news/ar`

AHIB A Hands-on Instruction to BPEL - `http://www.oracle.com/technology/pub/articles/matjaz_bpel`

OPM Oracle Process Manager - `http://www.oracle.com/technology/products/ias/bpel/index.html`

WPS WebSphere Process Server - `http://www-01.ibm.com/software/integration/wps/`

WID WebSphere Integration Developer - `http://www-01.ibm.com/software/integration/wid/`

ODE Apache ODE - `http://ode.apache.org/`

INTALIO Intalio Works - `http://www.intalioworks.com/products/bpm/opensource-edition/#overview`