WEB SERVICES COMPOSITION, EXECUTION AND VISUALIZATION

Manu De Backer, Guido Dedene, Jacques Vandenbulcke
K.U.Leuven, Department of Applied Economic Sciences,
Naamsestraat 69, B-3000 Leuven, Belgium
Manu.DeBacker, Guido.Dedene, Jacques.Vandenbulcke{@econ.kuleuven.ac.be}

Abstract

This paper presents a novel technique and a tool to Web service composition, execution and visualization. Web services composition refers to the act of interconnecting multiple elementary (or other composite) Web services, in order to create value-added functionality for customers. Graphical tools allow developers to define and create business processes on a "plug and play" basis. Runtime visualization of a business process enables the developers to understand the process and if necessary intervene in the process.

1. Introduction

This paper presents a short overview of Web service composition and describes a tool for business process development, execution and visualization. Web services are defined as a software system identified by a URI, whose public interfaces and bindings are defined and described using XML. Its definition can be discovered by other software systems. These systems may then interact with the Web service in a manner prescribed by its definition, using XML based messages conveyed by internet protocols [4]. One of the more interesting research areas in the Web services community today, deals with the problem how to interconnect several Web services in a larger composite system.

The remainder of this paper is structured as follows. First of all, Section 2 discusses the proposed system in detail as well as the different phases of a composite Web service architecture. In every subsection, we will especially indicate the potential and usefulness of program visualization for program comprehension. Subsequently, Section 3 gives a brief description of the demonstration. Finally, Section 4 concludes the paper.

2. Proposed system overview

Web services composition refers to the act of interconnecting multiple elementary (or other composite) Web services. These composite Web services offer organizations the opportunity to create value-added services for their customers. Several researchers and organizations are proposing standards to define the process of Web services integration (e.g. WSFL, XLANG, BPEL4WS, WSCL, WSCI, BPML, etc.)[2, 3]. Today, BPEL4WS is becoming the leading solution in this area.

The process of creating composite Web services consists of several sequential phases. Ranging from Web services **identification** and **registration** over Web service workflow or business process **specification**, to composite service **execution** and finally, **visualization**. The system is by no means finished and therefore we will clearly present at what stage we are now and were we would like to go in the future. The following paragraphs describe the different phases in more detail, together with the possibilities of using visualization techniques for program comprehension.

2.1. Web services Identification & Registration

Web services identification and registration refers to the process where the composite Web service developer needs to locate and register, if necessary, the elementary Web services he wishes to add in his business process. The most obvious technology to use is UDDI (Universal Description, Discovery and Integration), which is responsible in the service Oriented Architecture for the registration of the provided Web services. The directory service enables a look-up mechanism where consumers can go to find a service based on some criteria.

The identification and registration capacities of the tool have not yet been implemented, but it is our intention to implement this in the near future. We believe many interesting opportunities exist in this area.

Visualization techniques can be applied to create a graphical navigation structure of the UDDI registry. Users will be able to visually navigate the large amount of information in the UDDI registry, therefore they will be able to explore the registry and find what they need more efficiently.

Proceedings of the 12th IEEE International Workshop on Program Comprehension (IWPC'04)
1092-8138/04 $ 20.00 © 2004 IEEE

IEEE COMPUTER SOCIETY

2.2. Web services Composition & Execution

In order to create the composite Web service scenario, a graphical user interface (GUI) is provided. Using this GUI and its drag and drop features, a process developer defines a business process scenario and manually maps the output parameter of one service to one of the input parameters of the other. This data mapping is discussed in due course. While the business process is defined using the interface, the main instance of the process is stored in an XML based format.

Many problems arise when coupling Web services. In order to orchestrate several Web services and create a business process, output parameters and input parameters need to be interconnected. Therefore the data types need to be compatible and if not they should be converted. This process, also called mapping, is one of the more challenging in the Web services community today.

One of the problems of Web services that may prevent them from becoming the enabling technology in the e-business scenario, is the lack of reliability or Quality of service (QoS) they guarantee. In this context, we would like to especially mention the importance of availability and performance. Availability is a measure that indicates the readiness of the Web service to be invoked, while performance is the synopsis of latency, execution time, etc. Reusability has always been a delicate issue in software engineering, it denotes the degree to which a software module or other work product can be used in more than one computing program or software system [1]. Reusability will be implemented in the system by allowing previously created Web service compositions to be added to new projects.

2.3. Web services Execution & Visualization

The execution of the defined business process is handled by our execution engine, which is responsible for a wide set of tasks, ranging from code generation, type mapping, process verification, etc. The execution engine loads the scenario file, generated from our graphical editor, and then acts like a central mediator between all involved services and directs the corresponding message-flow

Our execution engine relies heavily on code generation. The engine receives an input scenario file, generated from the graphical editor. From this scenario file, the engine creates an executable version of the business process. Multiple options exist for this phase: a console application, a windows GUI application, or a new Web service. Each of these options should be implemented in our tool.

Exception handling, the process of handling run-time errors in a clean way, is also playing an important role. Catching exceptions is the first step, afterwards appropriate actions should be undertaken. Currently, our visualization tool uses a post-mortem visualization technique. This means that all the information, needed to visualize the business process, is gathered during the execution phase. Afterwards, when the system has executed the composite Web service, it loads the visualization file and generates the visualization. The next step in our visualization development would be a kind of runtime, live visualization of the system, which will then allow us to intervene in the execution process at runtime. Both techniques, post-mortem and runtime visualization, are called code-intrusive techniques, which means that they instrument the source code of the targeted program with special code that provides information to the visualization system.

3. Demonstration of the tool

Using a simple example, we would like to illustrate the key features of our composition tool. First of all, we will demonstrate the modelling of a business process using a set of registered Web services. In this phase, the process of data mapping will also be explained. Subsequently, the code generation step and visualization step will be demonstrated. Finally, we would like to show the tool's exception handling features.

4. Conclusions

In this proposal, we have presented the ideas and implementation of a Web services composition tool. This tool allows developers to easily create and compose Web services and additionally execute and visualize these composite Web services. Visual programming is used to interactively create a business process, while a post-mortem visualization technique takes care of the runtime visualization of the system. A central execution technique is used to execute the business process. We have implemented a prototype, using C# and the Microsoft .NET Framework.

5. Acknowledgement

This research is partially sponsored by Microsoft Belgium and Luxemburg.

References

[1] Institute of Electrical and Electronics Engineers. A compilation of ieee standard computer glossaries. *IEEE Standard Computer Dictionary*, 1990.

[2] W. van der Aalst. Don't go with the flow: Web services composition standards exposed, 2003.

[3] W. van der Aalst and A. Hofstede. Yawl: Yet another workflow language, 2002.

[4] World Wide Web Consortium http://www.w3.org, 2004.

IEEE COMPUTER SOCIETY