# IT Application Development with Web Services

**Christos Makris**
*University of Patras, Greece*

**Yannis Panagis**
*University of Patras, Greece*

**Evangelos Sakkopoulos**
*University of Patras, Greece*

**Athanasios Tsakalidis**
*University of Patras, Greece*

## INTRODUCTION

The advent of Web Services (WS) has signaled a true revolution in the way service-oriented computing and remote procedure invocation over the Web are conducted. Web Services comprise of a set of loosely coupled specifications to coordinate process execution from distance, based on common and widely accepted Web protocols such as HTTP, FTP, and XML, and therefore, providing increased development flexibility. Since the WS Framework was built on top of those protocols, Web Services have been widely acclaimed by the Web development community and paradoxically; they have marked one of the few examples in the history of computer protocols where a global consensus has been reached.

The Web Service framework consists of essentially three basic components:

1. The *Web Service Description Language* (WSDL), a language that allows formal functional characterization of the provided functionalities;
2. The *Simple Object Access Protocol* (simply SOAP from its version 1.2), a protocol that defines the format of the information interchange; and
3. The *UDDI* (Universal Description, Discovery and Integration) is a catalog of Web Service descriptions.

All three of these components are specified using XML markup. The elegance of the WS architecture lies in the fact that every WS transaction is taking place over established Web protocols such as HTTP and FTP. As remarked in Ballinger (2003, p. 5): "A Web Service is an application logic that is accessible using Internet standards." This very fact has accounted for the rapid and universal adoption of Web Services.

This work is organized as follows: First, a review of underlying technologies and tools is presented. Consequently, existing techniques for design methodologies are described. Next, an overview of storage and retrieval techniques for Web Services is given followed by real-world applications of Web Services. We conclude with open issues and discussion.

## BACKGROUND

The need for executing process from remote computers seems to have emerged right after the first networking efforts. To put it simply, people need to share their data or access other peoples' data over the Internet, in the easiest possible way (Ballinger, 2003, p. 2). These needs are formally described under the term *Service Oriented Architecture* (SOA). Typically, SOAs are distributed system architectures focusing on network-centric, message-based and platform-independent communication (World-Wide Web Consortium [W3C], 2004a, 3.1).

Web Services constitute a brilliant example of an SOA implementation, albeit not the only one. Some of its precursors include: UNIX RPC, Microsoft's COM/DCOM, CORBA, and Java RMI. All of the latter have failed however due to their complex architecture.

## REVIEW OF TECHNOLOGIES AND TOOLS

This section deals with the state-of-the-art in the technologies supporting the development of Web Services.

### The Web Service Framework

The base protocols for the Web Services architecture are HTTP, XML, SOAP, WSDL, and UDDI. The role of HTTP and XML is more or less self-explanatory; they provide the wrapper protocols for every kind of data communication. In

the sequel, we shortly describe the remaining protocols.

- *SOAP (W3C, 2003) defines the format of message interchange. This interchange takes place when* discovering, binding, consuming a Web Service.
- WSDL (W3C, 2004c), is a language to describe the functionalities of a Web Service and provide additional details about the ways it can be accessed, the points it can be reached, and so forth.
- UDDI (UDDI) defines a separate entity (registry) that mediates in the development process by hosting descriptions of Web Services.

First of all, SOAP (W3C, 2003) is a protocol that requires the creation of XML-like documents defining the format of every communication taking place during a Web Service deployment. SOAP messages are created each time a computer seeks or runs a WS and each time it sends messages, that is, query results, to a remote computer. A typical SOAP message contains data format, required WSDL messages for the procedure and, if it calls a remote procedure, the WSDL-coded names of functions that will be called.

WSDL (W3C, 2004c) is another WS standard built around XML specifications. While SOAP defines the format of messages that will be exchanged, WSDL is the language that totally describes Web Services. First of all, a WSDL document contains descriptions of data types, XSD or custom, used in the procedure invocation. The messages that must be exchanged used during the execution, that is conveying I/O information, are also defined. Names of available remote procedures are also registered. Furthermore, apart from the names, *bindings* are also defined; bindings refer to the message protocol (most commonly, SOAP and HTTP) in use and that is, to a URL that provides access to the service. An important feature of WSDL is the possibility to insert in arbitrary elements inside a WSDL document, meta-information to be utilized for documentation purposes.

Once having joined the Web Service game, one needs to find out which function to call to carry out a specific task, by which parameter, which protocol to use, and so forth. It was agreed during the early days of WS architecture to use centralized registries for this purpose. These registries would gather all the information about available Web Services they host and they would provide technical details, WSDL descriptions, on how to use them. These registries were termed UDDIs. The globally available UDDIs are few, including Microsoft's[1] and IBM's[2], however UDDIs also exist inside large corporations. The UDDI protocol requires the registry to provide the corresponding APIs for service registration and querying.

Corporate records inside a UDDI registry are implemented via *businessEntities.* Each party that wishes to publish a set of Web Services registers a new *businessEntity.* A *businessEntity* area contains business related information about the entity, which publishes the Web Services, such as contact information, e-mail, business categorization, and textual descriptions. A further sub-entity inside each *businessEntity* is the *businessService.* Each *businessService* contains a record for a conceptually or otherwise related subset of the provided Web Services. Web Services in the same *businessService* can be, for example, geographically related or performing different functionalities of the same category, e.g. functions

*Figure 1. The hierarchical structure in UDDI documents*