A FRAMEWORK FOR WEB APPLICATIONS DEVELOPMENT
A SOAP based communication protocol

Samar Tawbi, Jean-Paul Bahsoun
Institut de Recherche en Informatique de Toulouse, Paul Sabatier University, 118, route de Narbonne 31062 Toulouse, France

Bilal Chebaro
Lebanese University, Faculty of Sciences I, Hadath, Lebanon

Keywords: Generic Web development, Web services, SOAP, Communication Protocol.

Abstract: The rapid evolution of interactive Internet services has led to both a constantly increasing number of modern Web sites and to an increase in their functionality, which makes them more complicated to be built. In this context, we have proposed a generic approach for Web site development that manages the operational content of this kind of applications. A framework has been defined to support the development of web applications' processing tasks as Web services and the communication protocols with the users of these services. In this paper, we will expose the general structure of this framework, and we will focus on the communication protocol defined between the users and the system. Our approach in this protocol addresses universal clients; it is based on the SOAP protocol, XML language and their related technologies. It adopts the concept of Web services but uses it for providing code results rather than information results as it is known in the Web society.

1 INTRODUCTION

In the last few years the World Wide Web has been growing rapidly. The rapid evolution of interactive Internet services has led to both a constantly increasing number of modern Web sites and to an increase in their functionality, which makes them more complicated to be built.

In the early Web development practices, it was current for Web applications creators to simply build the solution, emphasizing in certain cases the development side of these applications (Ceri et al. 2000). But with the quick evolution of the Web and its related technologies on one hand, and the increasing requirements of users, on the other, many new features were implemented. And the Web applications became more dynamic, interactive and as a consequence more complex. This leads to a more difficult management and maintenance.

Many tools and technologies were proposed in order to increase web applications performance and the facility of their creation.

In this paper, we make an overview on the tools created for helping Web applications development and discuss how these products neglect to a certain level, development issues of these applications. We describe how we introduce the e-development concept to respond to this issue. So we define a framework that proposes a platform for managing operational functionalities for Web sites' creation. It gives a solution for the development difficulties by providing Web services for application development and management (Tawbi et al. 2002). This framework constitutes a basis for deploying servers providing development services in different Web domains. We will describe along the paper the communication protocol adopted in this framework that is based on Web services. But unlike what is usually done, the concept 'Web services' is used for providing development functionalities and not business services.

The paper will be organized as follows: In the section 2 we will talk about some previous work in term of tools for Web sites development and management and some of their shortcomings regarding the development side of Web sites. Section 3 will give an overview on the web services technology, its standards and how we could use it in

Tawbi S., Bahsoun J. and Chebaro B. (2005).
A FRAMEWORK FOR WEB APPLICATIONS DEVELOPMENT - A SOAP based communication protocol.
In *Proceedings of the Seventh International Conference on Enterprise Information Systems*, pages 82-89
DOI: 10.5220/0002537100820089

our work. We will outline our general approach in the section 4. Then, in section 5, we will give the structure of this framework and its properties especially the communication protocol that it uses. Section 6 will give an implementation example based on this framework. And finally, concluding remarks will be given.

2 WEB DEVELOPMENT TOOLS

A large variety of tools and technologies was created to support the publishing of static and dynamic data on the web. The first generation of this kind of tools was HTML editors (Elderbrock et al. 2002) (Lowery et al. 2001). Although, they helped the Web site creation by automating certain tasks concerning HTML code generation, they soon proved to lack a lot of dynamism and interaction. Web content managers (Aberdeen et al. 2002) (Kerer et al. 2002) (Vignette 2001), constitute the next generation of products for Web site engineering. They focus on offering sophisticated tools and architectures for delivering the information in a fast and scalable way. Their principle approach is to separate the content from the presentation of Web applications. From a technical point of view, although these products may offer component reuse, they do not rely on an explicit model. Their main problem is that they approach Web content management from the side of management and generation documents rather than the semantic of contents. Meanwhile, the research community has also worked on this problem and many models and systems were proposed like (Ceri et al. 1999) (Chauvet 2002) (Florescu et al. 1998) (Fernandez et al. 1999) (Gardner 2001) (Sadoski et al. 2000).

In general, all these tools take their idea from experiences in the database and hypertext fields. And they adopt for the data and for the Web site itself, structured data models very often close to relational databases, and a query language to communicate with the database. They focus the information content and address data intensive Web applications. This kind of systems have a primordial disadvantage that they don't take into consideration the operational and development aspects of Web applications especially complex ones.

In our approach we solve this specific issue by defining a framework that offers automated development tasks as online services, to the Web site creators. In other words it will be capable to provide services for Web code generation (HTML in a first step, and later XML). This framework separates completely the implementation from the presentation of applications. It defines a protocol for service management and a generic prototype for those ones in order for the framework implementations to be easily adaptable and upgradeable.

The next step was to choose a protocol of communication with the clients. We have found the concept of Web services the most appropriate for our problem. This issue is discussed in the next section.

3 WEB SERVICES TECHNOLOGY

To solve new difficulties that appear, organizations and enterprises try to define new technologies that are often very soon adopted by the wide public. In order to stay informed in the world of software development, one has to keep an eye on what is happening essentially in the big enterprises and all the new technologies used or created by those ones.

We know that the most important enterprises in the Web society are in permanent competition. Even though, these enterprises all admit that the concept of 'Web Services' will be very soon the principal and common language for business application.

Experts in this domain, have a future vision for web services in which there will be some kind of electronic services' 'cloud' on the Web; allowing users to choose anything they want for their applications. It's all the software industry that seems to be animated by some kind of conquest tendency for new virtual territories for economical exchange (Java Sun 2002).

The software applications, in general, are less and less considered as static programs executed on mainframes or PCs, but as applications functioning on a network of servers, located indistinctly in this 'cloud' of services, and that are generally accessed via Web interfaces. So, the design and implementation of Web applications became one of the major challenges for software industry in this century.

As its name shows, the term 'Web services' refers to accessing services over the Web. This term is now also used for the architecture, standards, technology and business models. In other words, it includes all techniques that make this possible(Gamma et al. 1995). IBM defines Web services as a new breed of Web application. They are self-contained, self-describing, modular applications that can be published, located and invoked across the Web. Web services perform functions, which can be anything form simple requests to complicated business processes (Vignette 2001). In other words, web services are interoperable building blocks for constructing applications.

Where the current web enables users to connect to

applications, the web services architecture enables applications to connect to other applications. Web services are therefore a key technology in enabling business models to move from B2C (Business to Consumer) to B2B (Business to Business) (Java Sun 2002). In addition, they present a black box functionality that can be reused regardless of their implementation. While technologies as DCOM (Brown & Kindel 2002), RMI (Elderbrock & Karlins2001) (Fernandez et al. 1997) and IIOP use object-model-specific protocols, Web services use normal Web protocols like HTTP and data format like XML.

Actually, as the researchers predict that XML will be soon the language of choice for Web development (Lowery et al.2001). In fact, the World Wide Web Consortium –W3C– created this language, in order to overcome the shortcomings of HTLM. Therefore, the relatively slow adoption of XML may refer to its complexity and the many related technologies and languages that must be used with it like XSL, XML Schema, DTD, etc. Furthermore, until now the XML based development tools are in their beginning if compared to HTML based ones. So it is important to create such a tool to support Web application development. For this reason, in our framework we have switched to XML. We adopted the concept of 'Web services' as well as its associated tools like SOAP (Brown 2001) (Ceri et al. 2000).

On another hand, even though the Web services concept was created and used originally to provide information or data result; we think it is possible to use it in our framework for code result corresponding to the operational tasks of Web applications.

4 A GENERIC APPROACH FOR WEB SITE DEVELOPMENT

Since the needs for sophisticated web sites is more and more relevant especially for the functionalities that are related to business domains like commerce, education and banking, and as we said that this domain is now adopted not only by computer experts but by the large public, we have proposed a generic approach for Web site development (Tidwell 2000).

It is a framework that offers a platform capable to provide the web site designers storage, access and processing of the information. It is independent of business domains and it defines a protocol for service creation. Servers created as implementation of this framework offer processing services in different Web domains.

The main idea is to approach Web site automated creation from the operational side and to separate the formatting tasks from the processing ones. Thus the framework defines the structure of servers that offer complex functionalities concerning the processing means in a Web site as interaction with users, database access, etc. Web applications creator could call these functionalities in their pages while remaining completely free in choosing any layout and formatting. The services to propose using this framework are specific to a certain domain on the Web like article publishing, emailing, e-commerce or others, unlike what we used to have in the automated Web page creation tools. Usually, these tools offer general-purpose services, leaving the task of specifying these services to the Web site developer. However, we focus in our work on more specific services in order to optimise their personalization task. For example, in an online article-publishing instance of this framework(Tawbi et al. 2002), we can find services for conferences or journal sites like paper submission, viewing, categorization, management, etc. This needs not only to define the development part of the functionality but to foresee also a certain data structure like database tables or XML files. Thus, we have complete independent services that could be used, with minimal personalization effort, in any Web application in the online article-publishing domain.

In order to have more flexibility, the framework offers two types of pages depending on the time when the service calls they contain are analysed and executed:

Dynamic pages are real time analysed, we mean when accessing the page by a certain Web browser. Service calls are added to the content of these pages when creating them, and they are hosted in this format. The processing of the services called is not done until the page is accessed by an end-user.

And Static pages that execute service calls at creating time. The result of these services processing is added to the content of the pages before they are published as any other page on the Web.

The figure 1 shows how the browser accesses the two kinds of pages. Dynamic pages are accessed via the service provider in our example it is the article-publishing server. Its role is to retrieve the page from the hosting server, to analyse it and to generate results for the service calls that it contains. This mechanism stays completely transparent to the end-user. In fact, there is no need to make any special processing from the navigator side to access these pages correctly.

On the other hand, static pages are directly accessed from the hosting server like any Web page without any additional processing effort.

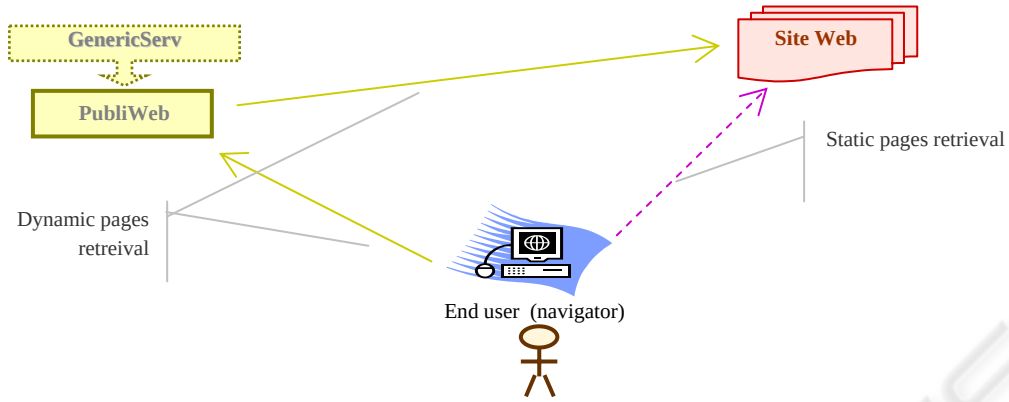

Figure 1: Accessing static and dynamic pages.

5 THE FRAMEWORK STRUCTURE

A principal concern in our framework is to reach a high level of genericity and independence from business domains. But at the same time, we must propose an architecture that allows easy instantiation and specification to any of these domains. After all, our aim is to propose development services for specific business processing tasks. Indeed, we must fix certain guidelines drawing the overall function of the framework and a common communication and management protocol. So we had to define a prototype for services' implementation independently of their application domain. This prototype represents a model of abstract classes that will be extended in order to create components representing concrete services. This allows easy specification to different Web domains.
Therefore, to create a service provider, for a certain field, based on the framework, we should implement the services needed in this field according to the proposed prototype and following the communication strategy adopted in the framework (figure2). At any time, server's developers can add a service component to their system in transparency of server users (Web site developers) and independently from the core work.

However, we may have some functionalities that are not specific to a certain business field like user authentification or registration to a mailing list for example. This kind of tasks is deployed in domain less services that constitute with the management and communication component, the server core representing the common part of these servers in any business domain on the Web. The services defined for a certain field constitute a layer over this common core.

This is true from the deployment and implementation point of view only. At this level of abstraction, we would like to point to the fact that the services are implemented and deployed in a late step, we mean after the framework has been defined with its common services, its communication and management protocols.
Architecturally speaking, the system has a three-tiers architecture (Shaw 1996) where we have a light client tier, a middle tier representing the service provider that is conceived in a layered architecture (Bahsoun et al. 2003) and a database tier (figure 4). In the next sections we will talk about these three tiers.

5.1 The Client tier

It represents the interface threw what users access the services offered by the provider. This is a thin client with almost no processing responsibilities; all you need is a browser to view the pages containing service calls. No need for any special component to be able to contact the server. In fact, all communications are based on Web standards like HTTP, XML, etc. this frees the web application developers from any programming effort.

5.2 The Processing tier

It contains the essential part of the services provider. It contains all processing, communication protocol and management, testing and security. Domain services will be developed in this tier based on the prototype defined by the framework. It has a modular loosely coupled object-oriented layered architecture (Bahsoun et al. 2003) in order to take all the advantages of the independence that this kind of architecture offers. So the framework architecture consists of four opaque layers (figure 3):

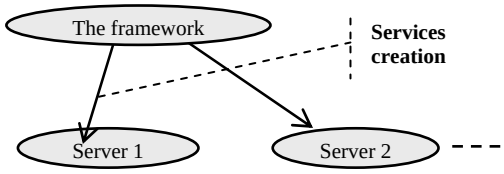

Figure 2: Specifying servers by creating domain specific services.

Connection Servlet: it is the interface of communication of the servers developed from this framework. It is a Java servlet that received users' requests representing Web pages to be analyzed. This connection layer represents the only gate to contact the server. In general, it communicates with the end-users' browsers, if we are talking about dynamic pages. Otherwise it is the Web site developer that contacts the server and in this case the communication is done via an interface for Web pages creation dedicated to this purpose.

Page Analyzer: this layer is responsible for analyzing the Web pages to extract service calls and sending them to the 'Service Dispatcher'. Page are analyzed in this layer at two levels, syntax level to verify if the call is well formed and semantic level to check for any incoherence in service calls, order and types of parameters for example.

Service Dispatcher: its job is to send service calls to the corresponding services after they have been analyzed. And in return it receives service results and sends them to the analyzer after it reformulates them in order to be conform to the format of the Web pages content.

Services Layer: this is the layer that will be specified to Web domains. It contains service management and the prototype for defining services if we talk from the framework side. But if we consider a server implemented using the framework, this layer contains the services offered in a certain domain in addition to basic services that are common to all domains.

5.3 The Database tier

It centralizes the data related to the applications in a repository database managed by a DBMS. It is accessed through the services offered to users, but stays completely transparent in order to allow independent updates and changes. It is accessed through an interface implementing the DAO pattern (Gamma et al. 1995). This makes the deployment of the Database server changeable and upgradeable independently of the services management from one side and of the clients from the other.

For this reason, we have standardized the management of the services and defined a communication protocol between the different actors of the application independently of the business domain of the server deployed. In the next section we will go more into the details of this protocol of communication.

6 A COMMUNICATION PROTOCOL

6.1 Introduction

The communication protocol to be adopted in our framework must respond to the constantly evolving needs. Besides, there is an important point to take into account is the genericity that we should preserve in all steps of our work including of course the communication protocol. This protocol should be independent of business domains in order to be applicable to all these domains when specifying the framework. Furthermore, when we talk about web server we address different types of users and by consequence different types of communication interfaces and devices as Wap phones or others.

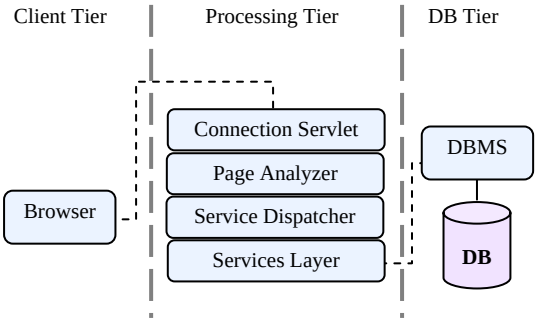

Figure 3: The system's architecture.

6.2 From HTML to XML based protocols

Several approaches were possible each one is based on a set of Web technologies and languages. However, it was necessary to adopt the most adequate one to our work taking into account all constraints previously exposed. Since we are working in Web applications development, HTML is the first language one can think to use for this field, as it is the most known standard. It is characterized by its simplicity and facility to use. One approach was based on this language where a Web site, using a service provider must be implemented in HTML. Knowing that HTML is text based, service calls consist of normal HTML text preceded by a specific keyword for example and inserted in the 'body' of the pages. Although, this way simplifies tremendously Web pages creation but it makes the page analysis process heavy and slow. Indeed, textual service calls require a textual parsing of the page, the HTML content will be analysed character by character in order to detect keywords indicating service calls. We can notice that by adopting this textual mode we loose a lot of performance and extensibility. These two characteristics are extremely important especially because we are working in the Web domain. In fact, the Web is wide spread this means thousands of users accessing the server. Besides, this domain is in permanent evolution from the two sides user needs and technologies. The use of a more extensible language at this stage seemed to be a must to preserve the genericity and the reusability of the framework. As we said in the previous section, XML offers an extensible way for representing information and it is now widely used. The free tags definition aspect that this language offers has allowed researchers to derivate mark-up languages for different domains as WML for WAP users, MathML for mathematics, and many others. For this reason, adopting XML makes the framework more flexible and adaptable to different types of users. Furthermore, XML parsers that have been specified by the Web community have proven their performance in comparison with textual parsers. Thus by adopting the XML language in our communication protocol, we could dispense with HTML browsers and use already specified XML parsers.

Now, we had to define a new protocol of communication between the Web applications and the service provider instantiated from the framework. This protocol should take advantage of strong points in XML language. Of course, we had to propose also a new service calls mode replacing the textual mode adopted.

In the following section we will describe this new approach, its functioning and its characteristics.

6.3 The functioning principle of the protocol

In general, using standard technologies in a certain system has two major benefits: first, the system is more universally accessible because standards are widely known. And second, the system inherits the advantages and the characteristics of these standard technologies and tools. This is due to the fact that these technologies are the result of many experiences' work in the domain. As we have seen in previous sections many standards were proposed for the Web at different levels and sides. As we are working on online service development, the 'Web Services' concept is very important for our new approach. Using Web services implies the use of certain related tools as the SOAP communication protocol (see section 3). The functionalities defined in our framework will be deployed as Web services using the SOAP protocol for communication. Usually a SOAP message consists of an XML document that contains a service call and its parameters (Ceri et al. 2000).

Therefore, in our system a Web page may contain many service calls in addition to normal content. Standard SOAP servers are not appropriated for managing this kind of documents. For this reason and in order to respect pages conceiving politic, we have considered a service call as a SOAP message but that will be inserted as an XML content in the Web document. We had to create a layer above the standard SOAP server representing a filter that is responsible of receiving Web pages and retrieving service calls for processing. The figure 5 shows this structure.

At analysis time, the filter extracts of the document all parts representing service calls and creates the corresponding message to send it to the SOAP server that in turn forwards the message to the appropriate service. We could import and use any SOAP server available like APACHE SOAP server or others.

6.3.1 The pages format

As with HTML, XML web pages in this protocol will contain normal content corresponding to the any information the Web site creator would like to put and service calls when necessary. Each one of these calls represents a SOAP message containing all required fields like SOAP Header and Body (Ceri et al. 2000) within a tag called 'ServiceBLock'. And normal content must be written within a

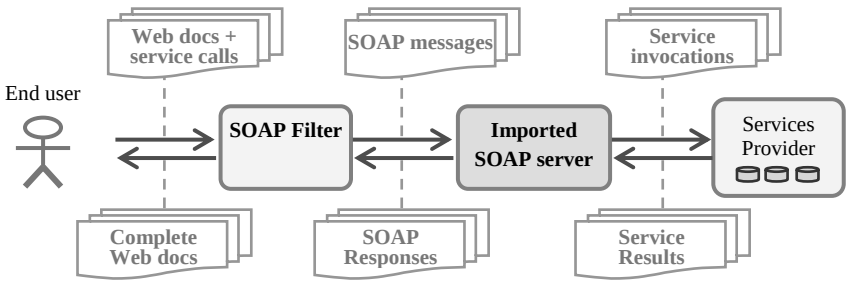

Figure 5: The SOAP filter constitutes a layer between the SOAP server and the users

'ContentBlock' tag. So, a page consists of a sequence of content and service blocks as in the figure 6.

So the pages will be hosted in the format previously described, surely this concerns dynamic pages only as static pages call services before they are hosted as we mentioned above. So, when analysing a page as in the example of the figure 6, blocks that correspond to XML text will be copied as they are to the response page and blocks representing a service call will be packed in a SOAP messages and sent for processing. In return, service responses are added to the page according to their order. Thus, we allow this way the Web site creator to have more of flexibility in managing the graphical format and design of the pages.

In fact, by adopting XML for page content, we make the servers compatible with more users in different types. However, as the XML is a free mark-up language, we should have a common language for the communication in order to allow the openness of the protocol. This issue will be discussed in the next section

6.3.2 A protocol for universal users

As we are dealing with an open language like XML and due to the many languages derived from this one. Web pages will contain then XML code according to a specific schema that is common to all servers implemented based on this framework. The content blocks of the request pages enclose information matching also to this same schema. This is because the analyser will not process this data, it will be sent as it is to the response page. So it must have the same structure of response page's content. By imposing a schema for pages definition we restrict the XML code of the Web developers to what is defined in this schema. However, this allows for the framework to propose a common format of the output data in the Web services and as a consequence unifies the pages analysing and the service processing.

According to this strategy, the response Web page is sent to the browser in XML format if this one supports the XML language. Otherwise, the server transforms the page to the language compatible with the browser depending on its type.

When a request is sent to the server, the SOAP filter receives it and detects the type of the browser WML, HTML or others. Then it send service calls to the SOAP server and to the service provider. Services produce result in XML then depending on the type of the browser the response page is either sent as it is in XML either translated to the browser languages according to XSL style sheets defined by the framework to each format (figure 7).

7 CONCLUSION

In this paper, we have presented a framework for Web application development. It offers a generic platform managing services that generate development tasks. This framework was created in

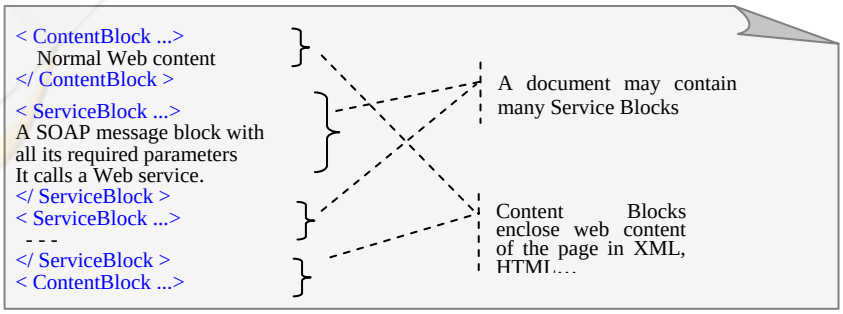

Figure 6: The structure of a Web document

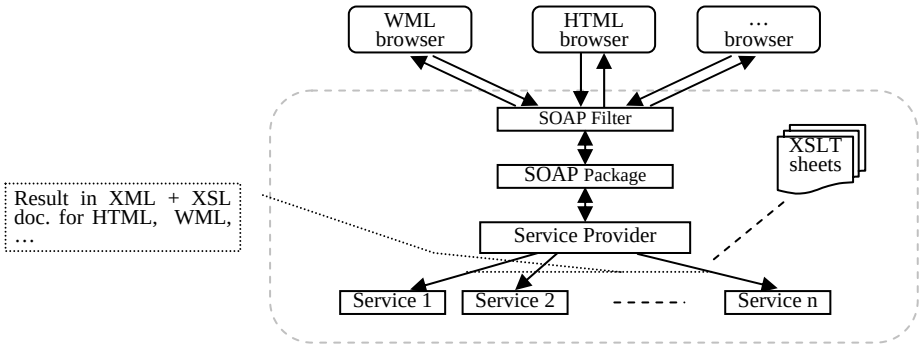

Figure 7: The response is sent to users depending on the type of the browser.

two steps. The first one was to build the structure of a server managing services in a generic manner in order to keep a high level of extension and evolution. So that this framework could be used universally, the second step consisted of adopting standard concepts and tools for the definition of a communication protocol like Web services, SOAP, etc. A layer was created over the SOAP protocol so that it could respond to the needs presented by the framework.

REFERENCES

Aberdeen Group Inc., 2002, *ColdFusion MX: Raising the Return on Investment of Internet Application Development*, White Paper.

Brown, N. & Kindel, C., 2002, Distributed Computing Object Model Protocol – DCOM/1.0, *www.grimes.demon.co.uk/ DCOM/DCOMspec.htm*.

Bill Brown, 2001, *SOAP Programming*, Sybex.

Ceri, S., Fraternali, P., & Bongio, A., 2000, Web Modeling Language (WebML): a Modeling Language for Designing Web Sites, In *IWWW00 the 9th International. World Wide Web Conference*.

Ceri, S., Fraternali, P., & Paraboschi, S., 1999, Design Principles for Data-Intensive Web Sites, SIGMOD Record : ACM Special Interest Group on Management of Data, 28(1).

Chauvet, J.M., 2002, *Services Web avec SOAP, WSDL, UDDI, ebXML*, Eyrolles.

Java Sun, 2002, http://java.sun.com/ blueprints/ corej2eepatterns/ Patterns/DataAccessObject.html.

Elderbrock, D. & Karlins, D., 2001, *Front Page 2002 Bible*, John Wiley & sons.

Farly, J., 1998, *Java Distributed Computing*, O'REILLY.

Fernandez, M., Florescu, D., Kang, J., Levy, A., & Suciu, D., 1997, STRUDEL: a Web Site Management System.

Florescu, D., Levy, A., & Mendelzon, A., 1998, Database Techniques for the World Wide Web: A Survey, SIGMOD Record (ACM Special Interest Group on Management of Data), 27(3).

Fernandez, M., Suciu, D., & Tatarinov, I., 1999 Declarative Specification of Data-Intensive Web Sites, In Proceedings of *the 2nd Conference on Domain-Specific Languages*, Berkeley, CA.

Gardner, T., An Introduction to Web Services, 2001, *http://www.ariadne.ac.uk/issue29/gardner/intro.html*.

Gamma, E., Helm, R., Johnson, R., & Vlissides, J., 1995, *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley.

Interwoven Inc., 2000, Application Development using Interwoven: Version 1.1, White Paper.

Kerer, C., Kirda, E., Krügel, C., 2002, XGuide - A Practical Guide to XML-based Web Engineering, In *the International Workshop on Web Engineering & Networking*, Pisa, Italy.

Lowery, J. W., & Lynch, K., 2001, *the DreamWeaver 4 Bible*, John Wiley & sons.

Sadoski, D., & Comella-Dorda, S., 2000, Three Tier Software Architecture, http://www.sei.cmu.edu/str/ descriptions/ .

Shaw, M., 1996, Some Patterns for Software Architectures, *Pattern Language Of Program Design*, Addison Wesley.

Bahsoun, J. P., Chebaro, B., & Tawbi, S., 2003, A Web Services Provider: Generic Architecture and Patterns for Business Applications, In IFIP03 *the third IFIP International conference on E-business, E-commerce, E-government*, Sao Paolo, Brazil.

Tawbi, S., & Chebaro, B., 2002, GenericServ: A generic server for web application development, In *the web requirements & e-services workshop of the 1st EURASIA conference for Advances in information & communication technology*, workshop proceedings, Austrian computer society.

Tidwell, D., 2000, http://www-106.ibm.com/ developer works/webservices/edu/ws-dw-wsbasics-i.html.

Vignette Corp, 2001, *Vignette Content Management Server*, White Paper.