

# Embedded Systems Integration Using Web Services\*

Guilherme Bertoni Machado  
Frank Siqueira  
Federal University of Santa Catarina  
Florianópolis, Brazil  
{bertoni,frank}@inf.ufsc.br

Robinson Mittmann  
Carlos Augusto Vieira e Vieira  
Boreste  
Florianópolis, Brazil  
{bob.mittmann,carlos.viera}@boreste.com

## Abstract

*Embedded applications, which were originally built on standalone devices, nowadays require a growing integration with other systems through their interconnection with TCP/IP networks. Web Services, which provide a service-oriented distributed architecture for the interconnection of systems through TCP/IP networks, have been widely adopted for the integration of business applications, but this sort of integration is still not provided by embedded applications. The present work aims to demonstrate the feasibility of using Web Services for the integration of embedded applications running on heterogeneous architectures. This is achieved through the provision of a support for the development and deployment of web services on embedded platforms. The feasibility of this approach is demonstrated by developing an application deployed on an embedded platform - the SHIP board - which is then integrated with a distributed enterprise application.*

## 1 Introduction

Nowadays, several applications use the Internet for the interchange of data (information) required for their execution. The Internet provides means for the integration of different applications through the use of standardized and widely available protocols that comprise the TCP/IP family. Nonetheless, the Internet does not provide semantic constructs for interpreting the data being exchanged between applications.

The problem of data exchange was partially solved with the creation of the World Wide Web [1] which was designed to allow the exchange of documents between computers using the Internet as its communication environment. Due to its ubiquity, the WWW was envisaged as the ideal basis

\*This research was supported by CNPq and FunPesquisa-UFSC

for the exchange of structured data between applications.

The Web Services architecture seeks to provide means of integrating applications by using open standards, protocols and languages widely adopted on the Internet, despite the intrinsic heterogeneity of the distributed environment.

Web Services are presented as an evolution of communication technologies based in Distributed Objects [2], which were the first successful attempt to provide a means of exchanging information between distributed applications in an heterogeneous environment. However, instead of using the object-oriented paradigm, Web Services adopt a Service Oriented Architecture (SOA) [3]. As shown in figure 1, a service broker behaves like a repository for the publication of the available service providers and for the location of these by service requesters. After being located, a service provider may have its services requested by a service requester.

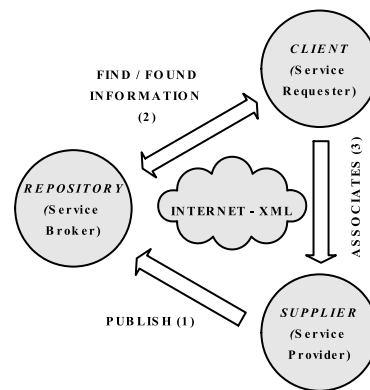


Figure 1. SOA Architecture

The main characteristics of Web Services are [4]:

- XML-based;
- Loose coupling between service providers and requesters;

- Coarse-grained;
- Client and service communication may be either synchronous or asynchronous;
- Supports RPC (Remote Procedure Call);
- Supports document exchange.

Moreover, Web Services employ the same communication protocols adopted by the Web, allowing data exchange through corporate firewalls. Web Services are also capable of providing applications and programs interaction without man intermediation [5]. Web services can communicate with other web services and then be able to supply new applications merging all their services. Web services are adopted mainly for enterprise application integration, but are almost unused in embedded platforms.

Embedded Systems currently lost their original meaning, which referred to small computational systems, usually isolated (stand-alone), which gives functional support for devices that do not fit in the definition of a computer [6]. We can define an Embedded System as a microprocessed device, therefore programmable, which uses its computing power for a specific purpose (for example, set-top boxes, microwave oven controllers, mobile phone chips, among others). These systems have some special characteristics [7]:

- Execute complex algorithms;
- User interface designed for a specific purpose;
- Real-time support;
- Multirate behavior (tasks that may be executed at different clock rates);
- Low manufacturing cost;
- Power management.

Embedded systems, comprise the biggest share on the integrated circuit market, and become more powerful every year with the evolution of the microprocessor technology. Embedded applications are also becoming more sophisticated, blurring the frontier between computers and other appliances. The growing similarity between embedded and desktop applications, enforces the necessity of interconnection of these devices through TCP/IP networks.

Knowing the potential of Web Services and analyzing the market trends of a bigger integration between different microprocessed appliances, there is nothing more natural than considering the use of web services in embedded systems.

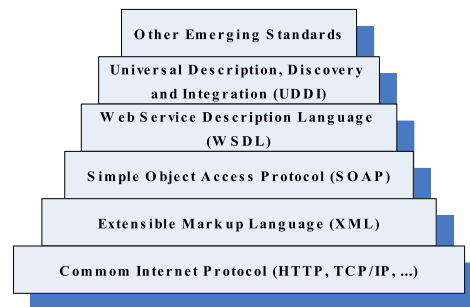
This paper presents our initial efforts to provide Embedded Systems integration through Web Services, describing

the design, modeling, implementation and deployment of web services built using the gSOAP development toolkit on an embedded system platform - the SHIP board.

This paper is organized as follows: In Section 2 and 3 are presented a brief overview of the major aspects (technologies and concepts) from Web Services and Embedded Systems and also a vision about how to structure Web Services in Embedded Systems. Section 4 lists some Embedded Systems boards and Web Services development toolkits and justifies the adoption of the gSOAP toolkit and the SHIP board (specification/characteristics of both are presented), and some related work found in the literature is described. Section 5 describes the proposed architecture for deploying web services on embedded platforms. A use case scenario is presented in Section 6, which is followed by preliminary results in Section 7. Finally, Section 8 summarizes our conclusions.

## 2 Web Services

Knowing that Web Services are not a specific technology, but a set of communication and (consolidated and/or emerging) interoperability protocols [8], we briefly describe the main technologies employed by Web Services.



**Figure 2. Web Services Layers**

As Figure 2, the core of the Web Service architecture is composed by an Internet protocol (HTTP in most cases), which sends encapsulated XML messages using the SOAP protocol. As superior layers we have the web services description language - WSDL and the repository that can be employed to publish and locate all web services - UDDI (Universal Description, Discovery and Integration). Beyond these layers, more recent studies aim to improve certain characteristics of the Web Services, such as: Security, Quality of Service, etc.

### 3 Embedded Systems

The block diagram of a typical Embedded System is represented by figure 3, these are the most common components in an Embedded System, where all uses some type of not-volatile memory (flash memory, EPROM or ROM, for example) and some form of RAM memory and I/O devices. Most Embedded Systems also have communications interfaces to connect to a development host.

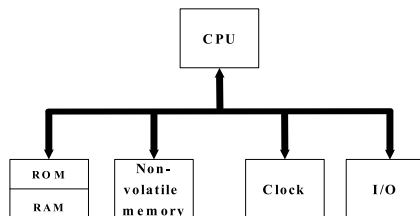


Figure 3. A typical Embedded System

### 4 Embedded Systems boards & WSDTK

There are several different embedded platforms in the market that are able to host a web service, which present the functionalities (native TCP/IP stack and HTTP support) necessary to integrate these devices using Web Services. For example: Netburner [9], RabbitCore (RCM family) [10] and SHIP [11], among others.

There exists a reasonable amount of Web Services development toolkits able to work in embedded systems; some are specific for embedded systems, while others have support for this kind of environment. We can enumerate the following toolkits: J2ME [12], ETTK [13], eSOAP [14] and gSOAP [15].

As main limitations for choosing a toolkit choice we point out the following characteristics:

- Hardware issues (processing, memory, more complex operating systems, etc...);
- Development environment (require commercial tools or a set of other softwares - as specific compilers);
- Incomplete, insufficient, unavailable or non existing documentation.

We have chosen the SHIP board and the gSOAP toolkit as tools to reach our research objectives based on a set of characteristics, functionalities and compatibility, such as: not only the development toolkit software but also the embedded platform operating system and the runtime libraries are open source and have good documentation and full support.

#### 4.1 SHIP

The first step to check the feasibility of the described ideas in this work consisted in acquiring a target platform (embedded systems board) in which is possible to deploy web services. Through an agreement firm with the company named BORESTE, we obtained the SHIP (development version) board, show in figure 4.



Figure 4. SHIP Board

This ARM based socket card presents the following characteristics:

- ARM7TDMI Microcontroller, family AT91X40 with RISC processor;
- 512K Bytes of flash Memory, with 448K Bytes free for the applications;
- Operating System developed by BORESTE which supports Ethernet connections and TCP/IP communications. Has as main characteristics the  $\mu$ Boot for system load and a debugger program -  $\mu$ Monitor - for configuration and load of the applications in the platform;
- Compact Web pages server (also developed by the company);
- Diverse external devices can be plugged in to the platform such as factory automation equipment, sensors and actuators, medical equipment, etc.

As technical characteristics of the platform we can enlist the low power consumption, a wide range of support libraries and some built-in software.

#### 4.2 gSOAP

In a second moment, it was necessary to port a Web Services development toolkit to the target platform. We have selected the gSOAP toolkit, because it has shown to be the most compatible with the chosen platform (SHIP only works with C routines and all the other development toolkits do not have this feature).

This toolkit possess the following characteristics [16, 17]:

- It makes the composition (binding) of SOAP messages with C/C++ creating Stubs and Skeletons;
- C/C++ clients through WSDL parser (WSDL converter to gSOAP header files);
- Platform independent (there are application examples developed in Windows, Linux, Unix, Mac OS X, Pocket PC, Palm OS, Symbian and embedded Linux);
- Applications can be released with less than 100K Bytes, with only 150K Bytes of total memory consumption.

Figures 5 and 6 (based on [16]) show how to develop and deploy services and clients using the gSOAP toolkit.

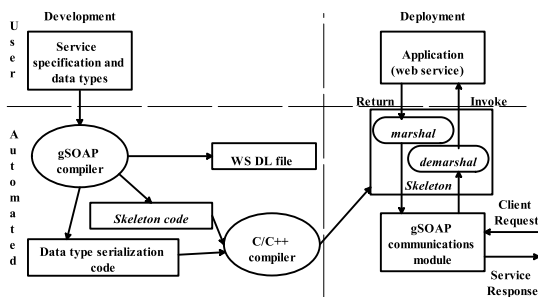


Figure 5. gSOAP Service Development and Deployment

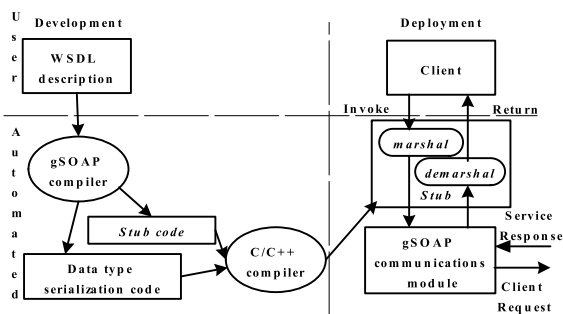


Figure 6. gSOAP Client Development and Deployment

It was necessary to make some changes in the source code in order to make gSOAP compatible with the SHIP operating system.

### 4.3 Related Work

There are few works in the area, and the existing works are described with little level of detail. The iPC (Internet Processor Chip) [18] is an embedded system built in a partnership between Samsung and Thinkware which already has the complete set of protocols for web services from TCP/IP to XML/SOAP.

Based in ARM7TDMI microcontroller, it uses the Wind River real time operating system (RTOS) developed by VxWorks. IPC has as main advantage the characteristic of being an integrated solution of hardware and software and as disadvantage we can cite the very poor documentation.

On the other hand, [17] and [16] describe the use of the gSOAP toolkit for web service development on more powerful Embedded platforms, such as Personal Digital Assistants (PDAs).

## 5 The Proposed Environment

This proposal is based on the environment illustrated by figure 7, where a client sends its XML/SOAP messages according to the required service type (previously known through the WSDL file located in the Embedded System and/or in a UDDI repository) through the HTTP protocol.

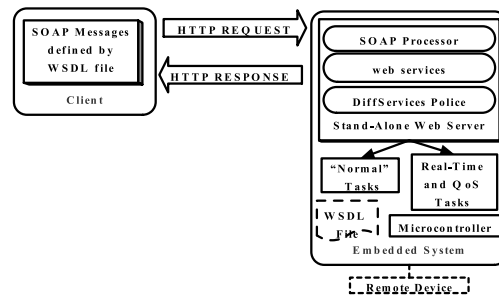


Figure 7. Environment

In the web service occurs a preprocessing stage, which consists in the efficient scheduling of requests based on their priorities - gold, silver, bronze and best effort. This policy tries to offers a fair chance that every request will be served based on the priority associated to each service.

Then these messages will be processed by the microcontroller. After the execution, a reply generated by the service may be returned to the client.

## 6 Use Case Scenario

The proposed architecture has been employed in an hypothetical scenario for supervising and controlling vital signs of hospital patients as shown by figure 8. Remote devices connected to a SHIP board send signals to acquisition equipment, according to preprogrammed parameters (real-time tasks have greater priority) and then this values are sent as XML/SOAP information to an application server.

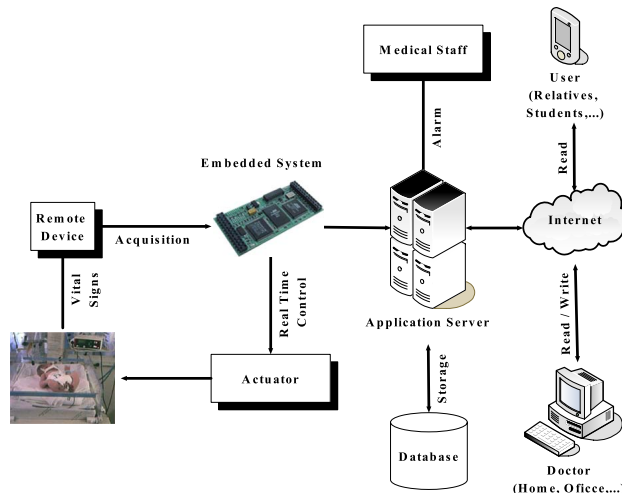


Figure 8. Scenario

The role of the application server is to store and distribute the diverse information coming from different sets of SHIP platforms and the attached remote devices inside the hospital environment to the respective clients (hospital database, medical staff at home or at the hospital, and patient relatives, using PCs, PDAs, mobile phones, etc).

In our example we can see clearly the services differentiation between services: gold for real time control, silver for alarms sent to medical staff, bronze for database update and best effort for other services.

The great advantage of this proposal is that, since we are using Web Services, the rest of the distributed environment has only to be able to handle SOAP messages. Concerns regarding integration with legacy systems, database structure, the kind of software the client is using, etc are irrelevant.

All these advantages depart from the principle that Web Services use XML over open architecture standards, therefore the use of this structure can be applied in the most diverse scenarios, such as:

- Supervision, Control and Diffusion of data in transmission lines (electricity, cable television, etc) and the most diverse systems/applications involved;

- Factory environments with the most diverse embedded systems with their specific communication interfaces.
- Network printers functionalities. Every function is a service and all of these can be access trough an stand-alone web server embedded in the network printer microcontroller.

## 7 Preliminary Results

At the present stage of this project, client-server applications were built using the platform to provide/execute the web service and a PC as client, demonstrating the possibility of deploying web services in embedded systems. In the near future we intend to develop more complex scenarios, involving a larger number of clients and servers to accomplish the integration of these with databases and applications servers, as described in section 6.

In order to reach our objectives, we must provide an environment able to dispatch requests with a reasonable response time to prove that the platform is able to replace a more powerful server platform such as a PC. Therefore, our initial tests began with few samples and threads with the objective to see how the SHIP board, especially its ethernet controller, react under a demand similar to the one it will experience in a scenario such as the one illustrated in figure 8.

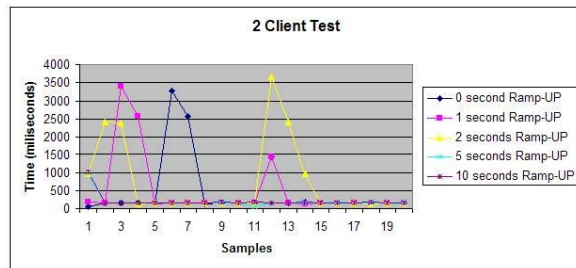
The service employed for testing purposes consists in a calculator service which has a total size of 65KBytes, including the operating system, the application server and the gSOAP libraries. So, when a client requests one of the available services (i.e., the four basic mathematical operations, implemented by methods `ns__add`, `ns__sub`, `ns__mul` and `ns__div`) a soap message is sent to the SHIP board, which performs the operation and replies with a SOAP message containing the result. After this, the server starts to wait for another service request.



Figure 9. 20 Samples 1 Client Test

As we can see in the performance measurements presented by figure 9, when a single client performs requests to the

services - a 464 bytes SOAP/XML-RPC request - the SHIP board has a 156 milliseconds average response time with a very small deviation between samples.



**Figure 10. 20 Samples 2 Client Test**

On the other hand, if more clients request services simultaneously - like figure 10, the behaviour observed shows that some requests have to wait longer while others are processed immediately. This enforces the need for a request scheduling mechanism such as the one described in section 5. These results were obtained using JMeter version 2.1.1 [19] running on a Pentium 4 2.8GHz with Windows XP and Java 1.4.2.

The first dispatched server with differentiation between services provides the simulation of two remote devices connection with an event policy based in which http port (80 or 81) is accessed. Port 80 has greater priority and requests sent to port 81 have lower priority.

So, when a client developed to call service A (connect to remote device 1, in this case a LED attached to IO port 17) a soap message is accepted. After five seconds this server disconnect the remote device and starts to wait for another service request (either A or B). Service B call is equal to service A except that connects another LED attached to IO port 18. If the SHIP platform is processing a call to service A, service B could be discarded.

Lots of efforts are under way to improve the provided software infrastructure. Scheduling algorithms, such as EDF (earliest deadline first) are being considered to be applied at application level as a first step achievement in a QoS policy and, therefore modifying the proposed scheduling mechanism.

## 8 Conclusions

Web Services present a way to interconnect applications through Internet among computational systems. Besides, its eminently open and standardized architecture provides to Web Services a great potential use in distributed computation.

Therefore, nothing more natural than proposing the deployment of a web services in a embedded systems to provide the integration of applications running on the Embedded platform with other system in a distributed environment. Starting with this study, we hoped to contributed in this new area of application of Web Services as middleware to Embedded Systems integration. Besides, this research intends to add QoS support in Web Services, because QoS requirements and its use politics are not still well consolidated in this technology.

## References

- [1] W3C. World wide web consortium. <http://www.w3.org/>.
- [2] W. Vogels. Web services are not distributed objects. *IEEE Internet Computing*, 7(6):59 – 66, Nov. - Dec. 2003.
- [3] J. Roy and A. Ramanujan. Understanding web services. *IEEE Internet Computing*, 3(6):69 – 73, Nov. - Dec. 2001.
- [4] D. Chappell and T. Jewell. *Java Web Services*. O'Reilly, first edition, Mar. 2002.
- [5] D. Lea and S. Vinoski. Middleware for web services. *IEEE Internet Computing*, 7(1):28 – 29, Jan. - Feb. 2003.
- [6] J. Janecek. Efficient soap processing in embedded systems. In *11th IEEE International Conference on the Engineering of Computer-Based Systems (ECBS)*, pages 128–135, 2004.
- [7] W. Wolf. *Computer as Components: principles of embedded computing system desing*. Morgan Kaufmann, 2001.
- [8] R. A. Kilgore. Simulation web services with .net technologies. *Proceedings of the Winter Simulation Conference*, 1:841 – 846, 8 - 11 Dec. 2002.
- [9] NetBurner. Netburner standard hardware. <http://www.netburner.com/>.
- [10] Rabbit Semiconductor. Rabbit ethernet connectivity. <http://www.rabbitsemiconductor.com/products/Ethernet/>.
- [11] Boreste. Ship - embedded ethernet board. <http://www.boreste.com>.
- [12] Sun Microsystems. Java 2 platform, micro edition (j2me). <http://java.sun.com/j2me/>.
- [13] IBM - International Business Machines Corporation. alphaworks emerging technologies toolkit. <http://www.alphaworks.ibm.com/tech/ettk>.
- [14] EXOR International Inc. esoap - embedded soap. <http://www.embedding.net/eSOAP/>.
- [15] GENIVIA Inc. gsoap - c/c++ web services and clients. <http://www.genivia.com/>.
- [16] R. van Engelen. Code generation techniques for developing light-weight xml web services for embedded devices. In *SAC '04: Proceedings of the 2004 ACM symposium on Applied computing*, pages 854–861. ACM Press, 2004.
- [17] R. A. V. Engelen and K. A. Gallivan. The gsoap toolkit for web services and peer-to-peer computing networks. In *CC-GRID '02: Proceedings of the 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid*, page 128. IEEE Computer Society, 2002.
- [18] WBC-Europe. Web services on a single chip. *PIM - Project Information Manual*, 2004. <http://www.wbc-europe.com/>.
- [19] The Apache Jakarta Project. Apache jmeter. <http://jakarta.apache.org/jmeter/>.