Integration of Embedded Devices Through Web Services: Requirements, Challenges and Early Results*

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.vieira}@boreste.com

Abstract

Integration of the currently available computing systems and platforms is one of the most envisaged goals achieved by computer scientists, which starts to become a reality nowadays. System integration technologies, such as Web Services, can provide a middleware for systems that were originally independent. This technology is being employed successfully for the integration of business software, allowing the interaction between systems run by different companies. Despite the high level of integration achieved in business to business interactions, the same integration is not always obtained at different levels within a corporation. The factory floor is an important example of lack of system integration. At this environment, several independent devices and communication systems coexist, and their integration often relies on customized solutions. Our goal in this paper is to investigate the adequacy of Web Services for the integration of the numerous devices that are found in a manufacturing cell in order to make these devices more easily reconfigurable and able to adapt themselves to changes in the production environment. This will be achieved by evaluating the support provided by a typical embedded platform and therefore identifying any limitation for its integration with other devices. Along the paper, we also propose changes in the firmware of these devices in order to allow their integration through the use of Web Services.

1 Introduction

Most of the devices that surround us nowadays - home appliances, mobile phones, audio and video equipment, PDAs and, of course, desktop computers - are not fully integrated, requiring them to be operated individually through their own user interfaces. Due to the availability of network connection between most of these devices though wired or wireless networks, it is possible for them to communicate with each other. But this is not all. Scientists are working on ways to allow them to *interact* - and not only to communicate - through standardized interfaces and communication protocols so that they can work in an integrated fashion. This will, in the near future, allow users to easily control these devices though any other device connected to the same network. This revolution was named "ubiquitous computing" by Mark Weiser in the early nineties [1], being also called "pervasive computing" nowadays.

Integration is also necessary between the different information systems employed by businesses. Aiming to provide means for easily integrating these systems, computer scientists have developed integration technologies such as CORBA [2] and Web Services [3]. These technologies have been successfully employed for the integration of information systems between different corporations with business ties, integrating the whole supply chain - all the way from the raw material provider to the product retailer. Integration of different information systems is also possible within a company through the use of Web Services, allowing different systems to exchange information, increasing the flow of information between different departments of a corporation.

The same degree of integration is desired within a company at a different level. The different devices that are employed in a manufacturing cell are often incompatible and unable to interact properly without the development of customized software, communication bridges and protocol converters. The addition of new equipment in the production cell or the manufacturing of a new product may require new software to be written, installed or configured. Awkward solutions are often employed for integrating and controlling these devices, resulting either in fully centrali-

*This research was supported by CNPq and FunPesquisa-UFSC

zed or in a hierarchical topology in which production equipment is connected to controllers that have to coordinate their operation. In hierarchical topologies, low-level controllers are controlled by their higher-level counterparts, which coordinate different production cells in the factory floor. Figure 1 illustrates these topologies. Both have several drawbacks such as limited performance, central points of failure and difficult maintainability.

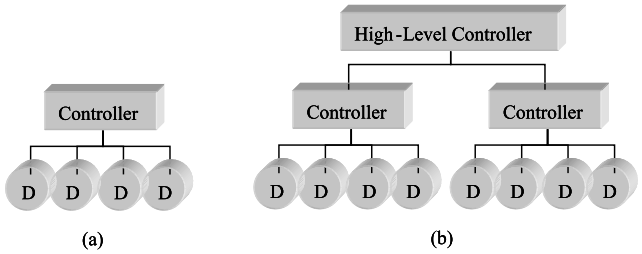

Figure 1. Centralized (a) and hierarchical control (b) of devices in a manufacturing cell.

Following the approach adopted for integration in pervasive computing and in business systems, this work aims to investigate ways of allowing fully independent manufacturing devices to interact without the necessity of employing controlling devices. This fully distributed approach aims to eliminate the previously described limitations of centralized and hierarchical architectures.

Limitations found in the software employed by manufacturing equipment are mainly related to integrated operation and management. Devices often are unable to interact without manual configuration, sometimes requiring specialized software to be written to allow interoperability. With the current diversity of equipment, such task is almost impossible to be performed. In order to overcome this limitation, scientists are looking for ways to allow device interoperation without human intervention.

The Web Services technology has been employed in order to allow the computer systems of different companies to interact naturally, due to the adoption of standardized ways of discovering and using the services that are provided by these systems. Furthermore, this technology adopts open, widely available communication protocols and a standard data format, avoiding complex protocol translations and data conversion along the communication path and at the target systems.

Our goal in this paper is to investigate the adequacy of the Web Services for the integration of the numerous devices that are found in production cells. This will be achieved by evaluating the support provided by a typical embedded platform and therefore identifying any limitation for its integration with other devices. Along the paper, we also propose changes in the firmware of these devices in order to allow their integration through the use of Web Services.

The following section describes the technologies employed by Web Services, which will be adopted for system integration. The architecture though which devices will be integrated will be described in section 3, together with a use case scenario. The following section presents the computing platform, the operating system and the Web Services toolkit that will be employed for device integration in the production environment. Section 5 presents the tests performed with the platform in order to evaluate the adequacy of web services for device integration. Related academic and commercial initiatives are described in section 6 and compared to the architecture proposed in this paper. Finally, the authors present their conclusions and perspectives for future work.

2 Integration Technologies

Integration technologies are intended to provide, at different levels, means of exchanging information between distributed applications in an heterogeneous environment. The most important of these are the Universal Plug and Play device architecture - uPnP [4], the Common Object Request Broker Architecture (CORBA) [2] and the Web Services architecture the main subject of our study.

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.

Based in a XML-based service-oriented distributed architecture (SOA) paradigm [5], Web Services provide the interconnection of systems through TCP/IP networks, which have been widely adopted for the integration of business applications. However, this sort of integration is still not provided at the device level due to the lack of support for this technology in most of these devices.

The core of the Web Service architecture is composed by an Internet protocol (HTTP in most cases), through which encapsulated XML messages are exchanged using the SOAP protocol [6]. As superior layers we have the web services description language (WSDL) [7] and the repository that can be employed to publish and locate web services - UDDI (Universal Description, Discovery and Integration) [8]. Beyond these layers, more recent studies aim to improve some characteristics of the Web Services, such as: Security, Quality of Service, etc [9, 10, 11].

Figure 2 illustrates the interaction between service consumers, providers and brokers.

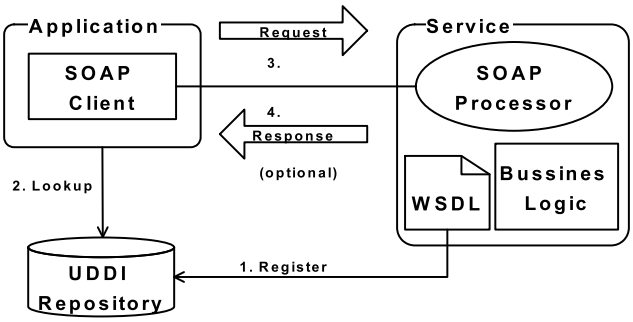

Figure 2. Web Services interaction

3 Architecture

Nowadays most of the embedded systems are capable of running standard Internet protocols such as TCP/IP and HTTP. Based on the support provided by these protocols and using a web services development toolkit suitable for embedded systems, we have proposed an architecture (Figure 3) for integrating embedded devices using Web Services [12].

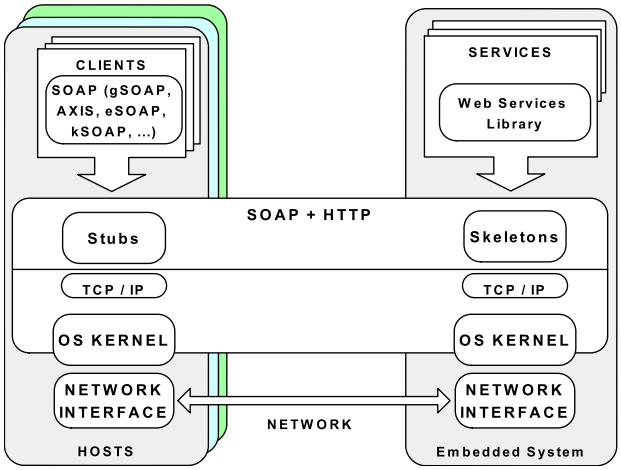

Figure 3. Environment Architecture

In our environment one or more hosts have clients which 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.

This massage reaches the Web Services library used to built the service(s) and then the method invoked by the client will be processed by the microcontroller. After the execution, a reply generated by the service may be returned to the client.

3.1 Use Case Scenario

The proposed architecture has been employed in an hypothetical scenario for industrial networks control systems integration.

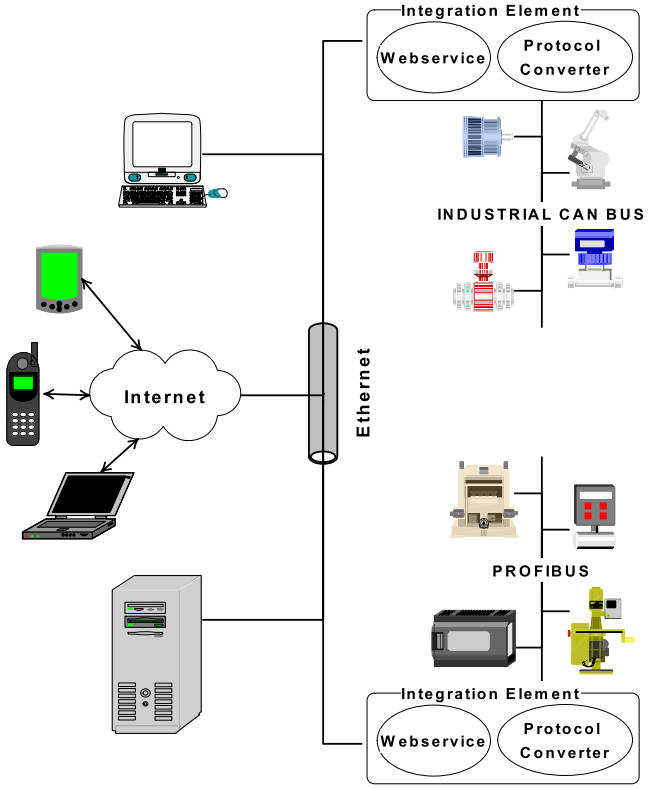

Figure 4. Scenario

As shown by figure 4, the proposed architecture can provide a support for the integration of heterogeneous industrial networks, such as Controlled Area Networks (CAN) [13] and PROFIBUS [14]. The integration element works as a protocol converter for each network bus and, for the other devices connected to the network, each industrial equipment will be seen as a Web Service. Therefore, a supervisory control and data acquisition PC for each network in the factory will be no more necessary, increasing the reliability, reducing bottlenecks and also eliminating the need for customized - and expensive - integration software.

In this scenario, SOAP messages will be exchanged between integration elements and its clients, while inside each industrial networks messages in network-specific format are exchanged. Each network implements its own control mechanisms, allowing the timing constraints imposed by each production cell to be fulfilled in a deterministic fashion.

Only high level control messages such as start, stop and

status are exchanged between integration elements and their clients, and whenever it is possible this messages are defined in WSDL files for each web service in the factory.

4 Development and Execution Platform

After searching for a suitable embedded platform and a Web Services development toolkit, we have chosen the SHIP board [15] and the gSOAP toolkit [16] for the integration of industrial equipment using web services. During this study we have employed the SHIP development version with a 33MHz ARM7TDMI Micro-controller and gSOAP version 2.7.3.

4.1 SHIP

Through an agreement firmed with Boreste (the company that manufactures the SHIP board) we have obtained this ARM based socket card which has built-in support for the TCP/IP protocol stack and an Ethernet interface. The SHIP can be plugged to diverse external devices, such as factory automation equipment. The SHIP board has a minimal operation system, but with enough features for deploying Web Services, leaving approximately 448KBytes of memory free for applications. The application server generated by gSOAP running a minimal web service takes approximately 65KBytes of memory. Additional services take as low as 2KBytes of memory each, depending on the complexity of the service. The Ethernet controller firmware of the SHIP board had to be reviewed to allow it to dispatch requests with a reasonable response time and to support a large amount of simultaneous requests.

4.2 gSOAP

Created by Professor Robert Van Engelen at the Florida State University, the gSOAP toolkit is able to deploy C and C++ web services applications and clients. As mentioned in [17], full SOAP interoperability is provided.

Using the soapcpp2 compiler we obtained a transparent binding between C and C++ data types and SOAP/XML data types. The generated file contains the stubs and skeletons to send messages using the SOAP protocol.

It was necessary to make some changes in the source code in order to make gSOAP compatible with the SHIP operating system. Some gSOAP definitions at configure.h, soapdefs.h and stdsoap2.h files were changed to adjust SHIP capabilities. Also some GNU libc constants portability issues were changed to values conforming to glibc-2.2.

5 Performance Tests

At the present stage, web services have been developed and deployed successfully in the SHIP board using gSOAP and have been tested using a desktop computer as client. Our preliminary results were obtained trough SOAP/XML-RPC requests using JMeter 2.1.1 [18] on a P4 2.8GHz with Windows XP and Java 1.4.2. We use the response time metric to analyze our environment.

The service employed for testing purposes consists in a calculator service. 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.

As we can see in the performance measurements presented by figure 5, when a single client performs requests to the services, the average response time is 156 milliseconds with a very small deviation between samples. On the other hand, if more clients request services simultaneously, the behaviour observed shows that some requests have to wait longer while others are processed immediately. All the SOAP/XML-RPC requests have 464 bytes.

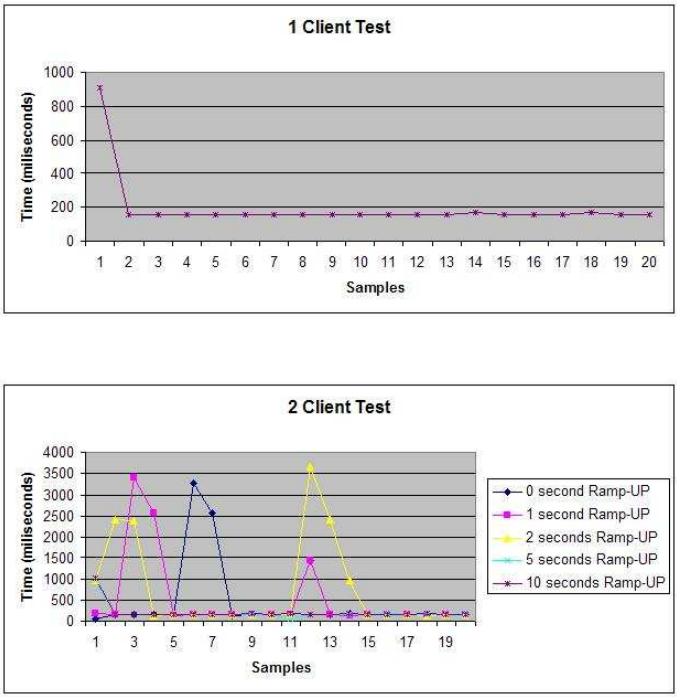

Figure 5. 20 Samples 1 and 2 Client Test

In a next stage, other tests, specially those who concern about memory usage, processing, transmission capacity and

priority between services will be made to achieve more detailed results.

6 Related Work

Related academic and commercial initiatives are found in the literature. Embedded platforms[19], [20], [21], [22] and [23], among others, present the functionalities (native TCP/IP stack and HTTP support) necessary to integrate devices in distributed environments through these embedded systems using Web Services. There also exists a reasonable amount of Web Services development toolkits ([24], [25] and [26], for example) able to work in embedded systems.

The related experiments found in the literature are described with little level of detail. In [27] is described the iPC embedded system which already has the complete set of protocols for web services from TCP/IP to XML/SOAP. On the other hand, [28] describe the use of the gSOAP toolkit for web services development on more powerful embedded platforms, such as Personal Digital Assistants (PDAs) and also cite gSOAP as part of software development packages for embedded systems.

Currently, as far as we know, there are no other viability studies or experiments with the same intent of the work described in this paper i.e., the integration of embedded devices through Web Services.

7 Conclusions and Perspectives

Integration of embedded devices through Web Services is possible, therefore the adequacy of Web Services for the integration of the numerous devices that are found in a manufacturing cell in order to make these devices more easily reconfigurable and able to adapt themselves to changes in the production environment can be done.

Using the Web Services paradigm brings lots of vantages such as:

- Modular and reconfigurable manufacturing cell. Instead of designing or buying a new and different supervisory control and data acquisition system for each industrial network, an integration element will work as a middleware to interconnect each manufacturing cell to the hole enterprise environment;
- Easy adopt - new devices, control flow, products and process;
- Cost - achieved trough open standards, protocols and languages widely adopted on the Internet and also, in our case, the development toolkit software and embedded platform operating system and runtime libraries are open code;
- Easy integration between factory floor and the other organization levels.

Our initial results demonstrate that the performance is viable to a significant set of applications (best-effort and soft real-time) and as a future work we intent to develop and deploy our ideas in a real scenario such as presented in this paper.

References

[1] M. Weiser. Hot Topics: Ubiquitous Computing. *IEEE Computer*, 26(10):71–72, October 1993.

[2] Object Management Group (OMG). Common object request broker architecture specification. Omg specification, OMG, Dec. 2005.

[3] D. Booth et al. Web services architecture. http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/.

[4] U. Forum. UPnP Device Architecture. http://www.upnp.org.

[5] J. Roy and A. Ramanujan. Understanding web services. *IEEE Internet Computing*, 3(6):69 – 73, Nov. - Dec. 2001.

[6] W3C. Xml protocol working group. http://www.w3.org/2000/xp/Group/.

[7] W3C. Web services description language (wsdl). http://www.w3.org/TR/wsdl.

[8] UDDI. Universal description, discovery and integration. http://www.uddi.org/.

[9] M. Conti, M. Kumar, S. K. Das, and B. A. Shirazi. Quality of service issues in internet web services. *IEEE Transactions on Computers*, 51(6):593 – 594, June 2002.

[10] D. A. Menascé. Qos issues in web services. *IEEE Internet Computing*, 6(6):72 – 75, Nov. - Dec. 2002.

[11] D. A. Menascé. Reponse-time analysis of composite web services. *IEEE Internet Computing*, 8(1):90 – 92, Jan. - Feb. 2004.

[12] G. B. Machado, F. Siqueira, R. Mittmann, and C. A. V. e Vieira. Embedded systems integration using web services. In *Proceedings of Fifth International Conference on Networking (ICN'06)*, Mauritius Island, April 2006. IEEE Computer Society Press. To be published.

[13] R. Bosh. CAN Specification Version 2.0. www.algonet.se/˜staffann/developer/CAN.htm.

[14] PROFIBUS International. PROFIBUS Technology and Application - System Description. www.profibus.com.

[15] Boreste. Ship - embedded ethernet board. http://www.boreste.biz/s.nl/sc.2/category.18/it.A/id.15/.f.

[16] GENIVIA Inc. gsoap - c/c++ web services and clients. http://www.genivia.com/.

[17] R. A. V. Engelen and K. A. Gallivan. The gsoap toolkit for web services and peer-to-peer computing networks. In *CCGRID '02: Proceedings of the 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid*, page 128. IEEE Computer Society, 2002.

[18] The Apache Jakarta Project. Apache jmeter. http://jakarta.apache.org/jmeter/.

[19] Ethernut. Open source hardware and software project for building embedded ethernet devices.

[20] ATMEL. Avr embedded internet toolkit.

[21] Unicoi Sytems Inc. Fusion web.
[22] NetBurner. Netburner standard hardware. http://www.netburner.com/.
[23] Lightner Engineering. Picoweb server.
[24] EXOR International Inc. esoap - embedded soap. http://www.embedding.net/eSOAP/.
[25] IBM - International Business Machines Corporation. alphaworks emerging technologies toolkit. http://www.alphaworks.ibm.com/tech/ettk.
[26] Sun Microsystems. Java 2 platform, micro edition (j2me). http://java.sun.com/j2me/.
[27] WBC-Europe. Web services on a single chip. *PIM - Project Information Manual*, 2004.
[28] 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.