BUILDING A WEB PLATFORM FOR LEARNING ADVANCED DIGITAL COMMUNICATIONS USING A MIMO TESTBED

L. Vielva, J. Vía, J. Gutiérrez, Ó. González, J. Ibáñez and I. Santamaría

Dept. of Communications Engineering. University of Cantabria. Spain
e-mail: luis@dicom.unican.es

ABSTRACT

Society demands access to high capacity wireless communications and to the services that can be provided on top of them. To satisfy these demands, engineers are constantly developing new technologies. These developments have to be selectively transferred to the university curricula. The student has to be familiar not only with the basic theory and techniques, but also with those of the more advanced techniques that provide a more profound insight. One of these techniques is based on Multiple-Input Multiple-Output (MIMO) systems. In this paper we show how to build a web platform for learning advanced digital communications based on a MIMO testbed. Using as starting point a 4×4 flexible dual band ($2.4/5$ GHz), we first develop a webservice interface to provide remote access to the services offered by the testbed and then implement a virtual laboratory where the students can parameterize and perform advanced experiments.

Index Terms— Communication engineering education, MIMO systems.

1. INTRODUCTION

In a previous work, we presented a DSP-based Digital Communications laboratory, based on universally available and low-cost hardware, that uses a problem-solving approach to learning communications systems from the very beginning [1]. The student of more advanced courses has to be familiar not only with the basic theory and techniques, but also with those of the more advanced techniques that provide a more profound insight. One of these techniques is based on Multiple-Input Multiple-Output (MIMO) systems. In this paper we show how to build a web platform for learning advanced digital communications based on a MIMO testbed. In Section 2 the hardware testbed and the application program interface (API) provided by the manufacturer are described. In Section 3 the requirements and design of the web platform are described and the developed architecture is presented. In Section 4 the functionality of the developed system is described, and a fully functional learning example is presented.

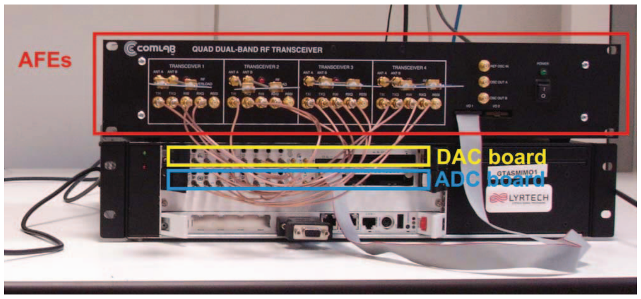

Fig. 1. One of Lyrtech nodes: with the PC, transmitter and receiver at the bottom; and RF front-end at the top.

2. DESCRIPTION OF THE 4×4 MIMO TESTBED

The MIMO testbed consists of two Lyrtech nodes as the one shown in Figure 1. Each node consists of a personal computer (PC) with Microsoft Windows XP operating System, transmit (DAC) and receive (ADC) boards, and an analog radio-frequency (AFE) front-end which performs the up/down conversion to base band or to intermediate frequency. The manufacturer provides a static library (LIB) and a header file that implement an API to develop programs using C as a programming language. This is the native way of controlling the hardware from user applications running locally on the PC of the node.

The testbed supports real-time wireless transmission, allowing the student for experimenting with real physical channels. Algorithms can be implemented and tested online, since the equipment makes use of Field Programmable Gate Array (FPGA) technology for real-time processing of the baseband signals. The FPGA of the transmitter and receiver nodes are interfaced with conversion boards of eight 14 bits converters. The AFE, with four independent receive/transmit channels, is linear over a very large dynamic range of the signal. It is entirely based on the MAX2829 single chip RF transceiver, which has been specifically designed for Orthogonal Frequency Division Multiplexing (OFDM) 802.11 WLAN applications. It covers the Industrial, Scientic and Medical (ISM) frequencies of 2.4–2.5 GHz and 4.9–5.875 GHz and supports up to 40 MHz channel bandwidth. The MAX2829 integrates

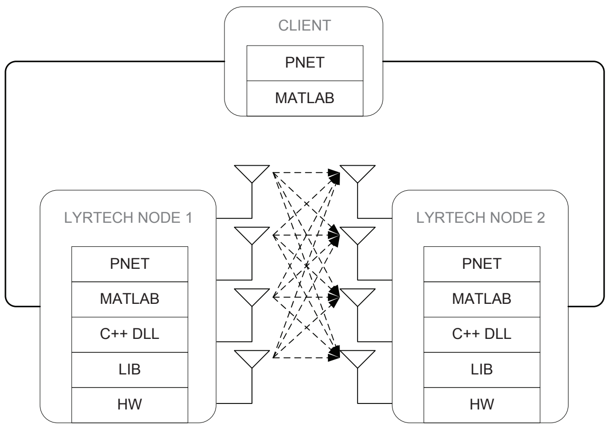

Fig. 2. The first step consisted of providing access to external applications, like Matlab, running on the node PC to control the testbed. By using pnet, access was given to remote Matlab instances.

all the circuitry required to implement the RF transceiver function, providing a fully integrated receive path, transmit path, Voltage-Controlled Oscillator (VCO), frequency synthesizer, and baseband/control serial interface.

3. DESIGN OF THE WEB PLATFORM

In order to implement an interactive web laboratory, we first performed a bottom-up feasibility study, identifying just one critical point: the ability of external applications to interface with the nodes, had to be verified (sending and receiving data as well as commanding the nodes). To validate this critical point, we developed a C++ dynamic link library (DLL) that encapsulated the LIB provided by the manufacturer and allowed external applications running on the node's PC to control the hardware, as shown if Figure 2. At this point, we were able to use Matlab in all the stages of the MIMO communication process: to prepare the data to be transmitted, send the data to the transmission node, control the sending and receiving process, get the data from the receiving node, analyze the data, and display the results.

In spite of the great functional advance obtained by just dynamically linking the nodes with Matlab, it was not close to be a good environment for educational purposes because of three main problems: (i) the configuration required Matlab to be installed and running on each of the testbed nodes; (ii) the students would need to have full access the node's PC, both being a security risk and limiting the experiments to local interaction; (iii) the fact that Matlab provided full access to the builder's API was two-sided, since the availability of functionality implied extra care to be taken interacting with the nodes. The necessity of students having local access to the node's PC could be relaxed by using pnet, a Matlab toolbox that allows a local Matlab (the student PC) to control a remote Matlab (node's PC). However, this remote access technology was far from ideal and all the other caveats remained.

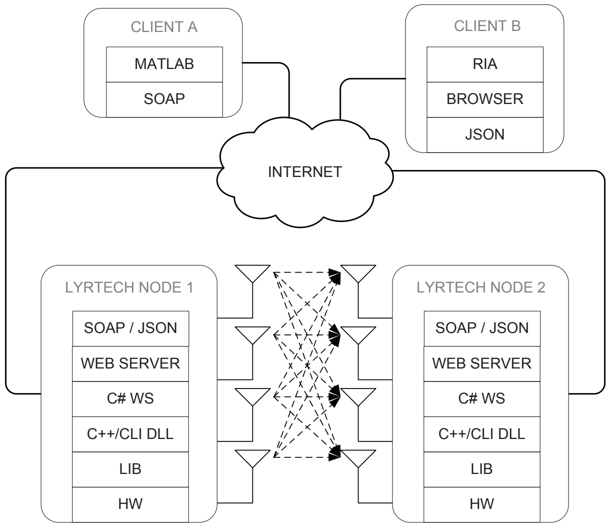

Fig. 3. Architecture of the web platform. The nodes provide a dual webservice interface. Client A is a Matlab application using SOAP interface and Client B is our web application using JSON interface.

We identified a set of requirements for our educational web platform:

- To provide a well defined interface so that the applications have access to the required functionality and have no chance to damage the node.
- To isolate the communications interface from the application program.
- To maximize interoperatibility with other programs and third party tools.

The requirement about the application programs is very important: on the one hand, we wanted an integrated web platform to provide the students with well-designed experiments and an interactive and responsive environment; on the other hand, we wanted that third-party applications could interact in a controlled way with our hardware nodes. In particular, we wanted to give good support to Matlab users.

In order to satisfy these requirements, we designed an architecture with three different layers on the server side: (i) a Microsoft .NET library developed in C++ as an interface with the builder's LIB; (ii) a set of classes developed in C# that represented the nodes as objects with properties and supported methods; (iii) a set of webservices that provided the only access of the external world to the nodes. On the client side we have two scenarios: (i) we used a wrapper of the webservice as a Matlab class that provided access to those users and experiments that wanted to use Matlab as an application program; (ii) we designed a web application using international standards and recommendations that implemented the set of experiments defined for the virtual lab.

For the webservice layer we developed a dual interface: (i) a Simple Object Access Protocol (SOAP) interface that is a World Wide Web Consortium (W3C) recommendation [2] that allows third party applications, including Matlab, to transparently and safely interact with our nodes through the provided interface; (ii) a JavaScript Object Notation (JSON) interface [3] to the same node API for been used by our own web application. The reason for this dual interface is that both methods are widely used, and are best suited to different scenarios: JSON has less payload and is more appropriate when the application program is a JavaScript application; SOAP is more verbose but is supported by a broader range of application programs. In both cases, for sending binary data, they are first encoded using base64 in a similar way as current mail applications encode the attached photographs [4].

The global architecture is shown in Figure 3. At the bottom are the two Lyrtech nodes, consisting of the MIMO hardware and the supporting PC, are shown. The ascending layers correspond to hardware, manufacturer's LIB, C++ wrapper DLL to gain access from managed code, C# classes that implement the nodes as objects and provide dual interface, the web server and the dual interface provided in SOAP and JSON. At the top two different clients are shown: client A represents a remote Matlab application that interfaces with the MIMO testbed through the SOAP interface; client B represents our web application that uses the JSON interface.

Current technology makes it possible to develop software applications that integrate text, animated graphics, video and audio to provide a multimedia exploration environment. Traditionally, there has been a dichotomy between desktop and web applications. On the one hand, desktop applications were able to provide a much richer multimedia experience. On the other hand, web applications are much more convenient for use as a learning tool due to the ease of deployment, updatability, and universal access to the students. Recently there has been a number of technological advances that have made it possible to develop interactive multimedia web applications, known as rich Internet applications, with all the richness traditionally reserved for desktop applications that run natively on last-generation web browsers without using Java applets, ActiveX controls or any third party plug-in like Adobe Flash Player.

Our educational tool uses the following key technologies and standards: HTML (Hypertext Markup Language) for the textual user interface like buttons, lists and text; JavaScript as the programming language that runs natively on the web browsers and modifies dynamically the user interface; SVG (Scalable Vector Graphics), a W3C recommendation [5] for describing two dimensional graphics in XML (Extensible Markup Language); and Ajax (Asynchronous JavaScript and XML), a set of technologies to implement communication between the browser and the web server.

4. VIRTUAL LABORATORY

As an example of the educational uses of the web platform, we describe a beamforming experiment in a OFDM MIMO 4×4 ($n_{\text{TX}} = n_{\text{RX}} = 4$) communication [6]. The experiment consists of the following stages:

1. Node setup: the FPGA's are programmed and the parameters (transmission or reception role, transmitted power, operation frequency, ...) are set.

2. Training stage: a node transmits a training frame that is received by the the other node, as show in Figure 4. The training frame consists of $N = n_{\text{TX}}n_{\text{RX}} = 16$ OFDM training symbols that are transmitted and received under different beamforming vectors. The application allows using any orthogonal set of beamformers to perform the estimation of the MIMO channel. In this example, we are using the columns of the identity matrix (only one pair of Tx and Rx antennas is active during the transmission and reception of each of the OFDM training symbols).

3. Optimal beamforming selection: the application allows the student to analyze the received frame in order to perform Least Squares MIMO channel estimation. Figure 5 shows an example of the frequency response of the $N = 16$ equivalent SISO channels that constitute the 4×4 MIMO channel. The student is able to compute the optimal beamformers according to certain criteria (to maximize signal to noise ratio (SNR), to minimize bit error rate (BER), ...).

4. Data transmission: data is transmitted and received with both the optimal and default beamformers, as shown in Figure 6. The student is able to realize of the improvement obtained in terms of received power, error vector magnitude (EVM) of the constellation or BER.

5. CONCLUSIONS

We have successfully implemented a web platform for learning advanced digital communications using a MIMO testbed. The platform is quite versatile: on the one hand allows third-party applications, like matlab, to fully interact with the testbed in a controlled way. On the other hand, specifically designed web applications, like our own rich Internet application, can be developed to provide interactive experiences as a virtual lab for the students.

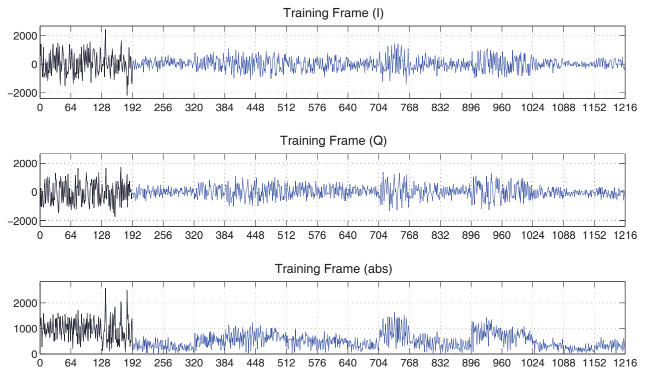

Fig. 4. Real (top), imaginary (middle), and magnitude (bottom) values of the time-domain received training frame.

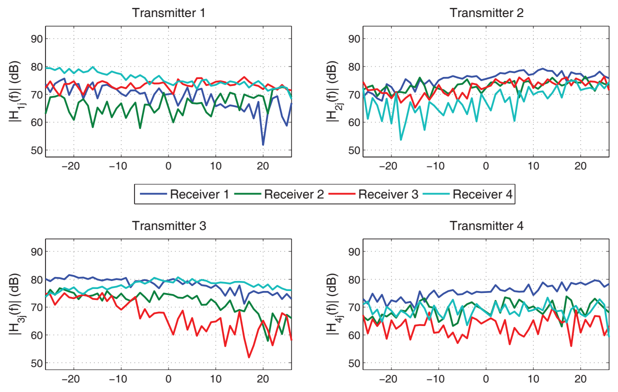

Fig. 5. Magnitude of the frequency response of the 16 estimated equivalent SISO channels of the 4×4 MIMO channel.

It is worth pointing out that the developed web platform, not only allows the student to perform offline signal processing tasks —using Matlab, for example— on the data received and recorded by the nodes, but it also allows the student to perform real-time processing on the platform by designing and developing algorithms with system generator and uploading the resulting code to the FPGA's of the nodes.

6. ACKNOWLEDGMENTS

This work has been supported by Spanish Ministry of Education and Science (MEC) under project "MIMO technologies for multiterminal wireless networks (MultiMIMO)" TEC2007-68020-C04-02/TCM and grant BES-2008-002085, and by the European's Community Seventh Framework Program under grant agreement n 213952, "MIMAX".

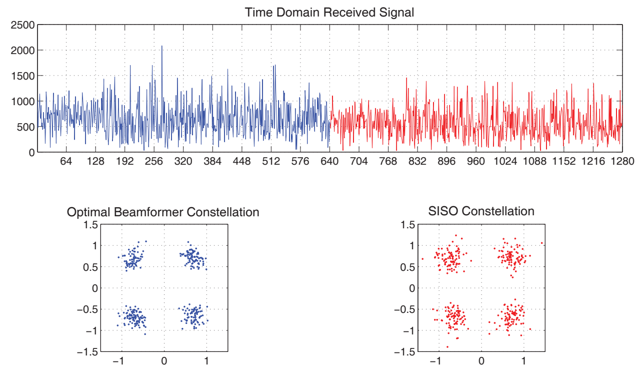

Fig. 6. Time domain (top) and constellations (bottom) of the received frames with both optimal (red) and default (blue) beamforming.

7. REFERENCES

[1] J. Ibáñez, C. Pantaleón, L. Vielva, and I. Santamaría, "Teaching digital communications: A DSP approach," in *IEEE Int. Conf. on Acoust., Speech, and Signal Processing (ICASSP 2003)*, Hong Kong, China, April 2003.

[2] Yves Lafon and Nilo Mitra, "SOAP version 1.2 part 0: Primer (second edition)," Tech. Rep., W3C, Apr. 2007, http://www.w3.org/TR/2007/REC-soap12-part0-20070427/.

[3] E. C. M. A. International, *ECMA-262: ECMAScript Language Specification*, ECMA (European Association for Standardizing Information and Communication Systems), Geneva, Switzerland, third edition, December 1999.

[4] S. Josefsson, "The Base16, Base32, and Base64 Data Encodings," RFC 4648 (Proposed Standard), Oct. 2006.

[5] Jon Ferraiolo and Dean Jackson, "Scalable vector graphics (SVG) 1.1 specification," W3C recommendation, W3C, Jan. 2003, http://www.w3.org/TR/2003/REC-SVG11-20030114/.

[6] J. Vía, V. Elvira, I. Santamaría, and R. Eickhoff, "Minimum BER beamforming in the RF domain for OFDM transmissions and linear receivers," in *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP 2009)*, Taipei, Taiwan, April 2009.