Wireless Networks In-the-Loop: Creating an SDR Development Environment

Nico Otterbach, Martin Braun and Friedrich K. Jondral
Communications Engineering Lab, Karlsruhe Institute of Technology (KIT), Germany
Kaiserstr. 12, D-76133 Karlsruhe, Germany
Email: nico.otterbach@student.kit.edu, martin.braun@kit.edu, friedrich.jondral@kit.edu

Extended Abstract

***Index Terms*—Software Radio, GNU Radio, Wireless Networks, Network Simulation**

I. Introduction

With the expanding availability of powerful software defined radio (SDR) platforms, the way of developing wireless systems has changed. A modern SDR framework offers the opportunity to do the entire baseband signal processing in software. Thus, typical hardware design problems and costs are circumvented.

Another advantage of this trend is the possibility to shorten development cycles by applying software-based development methods such as test-driven development.

Development costs and time can be decreased even further by applying the principle of software-based testing to the development process of entire wireless networks instead of just considering the in- and output of a single nodes. Furthermore, the tools provided to the developer need to change the way the development process has changed.

A first attempt to reach this goal was made in [1]. The introduced system, called *Wireless Networks In-the-Loop (WiNeLo)*, is based on the GNU Radio [2] software development toolkit and supports the measurement and simulation of transmission channels. It works quite well for simple nodes, but has to be heavily extended in the case of more complex, transceiving or adaptive node. Moreover, the development process as such has not even been taken into account so far.

In this paper, we will focus on the lack of development and testing tools and demonstrate how to simulate more complex wireless networks using the WiNeLo framework.

The basic idea is to provide the developer with a central control interface allowing him to run automated, pre-defined test scenarios in both the simulation and the real-world without needing to make adjustments to the code. Therefore, the testing- and the production code is the same, not having a layer of abstraction between them, which leads to more realistic results during testing. Additionally, there is no need to create extra code for more abstract model-based analysis.

To model the physical influence of the transmission channel, we will make use of the existing WiNeLo features. In addition to that, generic blocks were implemented to face some basic influences introduced by the RF hardware.

Some new possibilities, the achieved performance and a typical development cycle as seen in **Figure 1** will be presented within the scope of the demonstration.

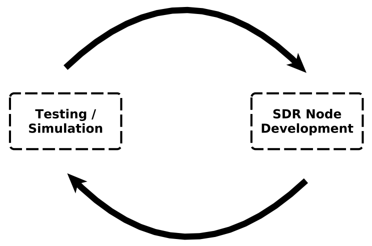

Fig. 1: Typical development cycle

II. Wireless Networks In-the-Loop

The basic principle of Wireless Networks In-the-Loop is shown in **Figure 2**. The GNU Radio applications are connected to a simulation sink or source which is controlled by a simulation server. The raw I/Q-samples are exchanged over the network to offer the possibility of simple load balancing. In addition, this allows us to move the computationally expensive server-side channel and hardware emulation to a more powerful machine.

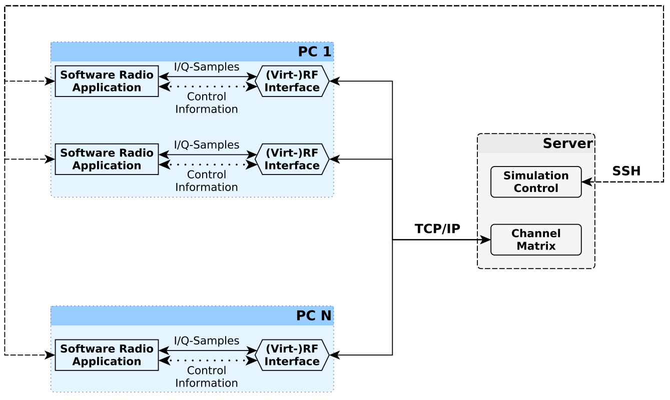

Fig. 2: Basic principle of WiNeLo

The physical influences and non-idealities between the digital-to-analog and the analog-to-digital converters can be separated into two major sections:

- The transmission channel: This topic was covered by [3]. As a result, WiNeLo offers a channel measurement and simulation tool.
- The RF frontend: [4] focused on this aspect. In addition, WiNeLo comes with some basic blocks to model the most relevant effects like frequency offset, phase noise, etc..

The physical influences are already well modelled within WiNeLo, but to serve as a development environment a lot of features, like accurate modelling of the timing behavior, rate-conversion and mixing, had been missing so far.

Some of the required enhancements will be covered in the next section.

III. Novel Development Techniques

A. Seamless Switching

To circumvent the need of abstract models with extra code in the early development phase, it is necessary that the production code can be used for both the simulation and later real-world, hardware-based tests. This is achieved by reproducing the interface to the hardware. Hence, our simulation interfaces offer exactly the same in- and output behavior as the "Universal Software Radio Peripheral" Hardware Driver (UHD) [5] block in GNU Radio does.

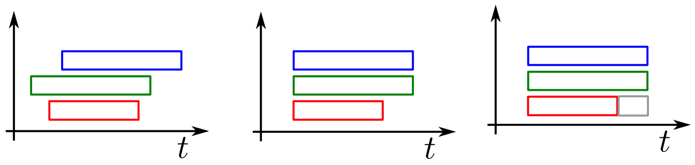

Fig. 3: Sample alignment and insertion of zeros

Apart from an identical interface, the simulation needs to rebuild the timing behavior in an identical manner. This is especially of interest if you want to simulate dynamic medium access as it will be used by adaptive or cognitive radios.

Therefore it is necessary to be able to correctly align, overlay and synchronize multiple streams of samples. **Figure 3** shows the situation at the server. The samples arrive at different times caused by network delay, so the server has to align them correctly. Afterwards, one of the streams needs to be extended in order to ensure synchronicity. This situation can occur if a transmitter stops sending samples, e.g. in a TDMA system. The server then has to fill the gap with the correct amount of zeros.

B. Centralized Test-Management

In response to the changed development process of wireless systems, new development tools should be provided to the developer.

Centralized test-management enables the developer to automatically control and parameterize all involved subsystems of a pre-defined testcase from a central user interface. This includes starting and stopping GNU Radio applications as well as pausing them as an instrument to debug the entire system. These new capabilities additionally offer the possibility to run automated tests with a single passed/not passed result. Additionally, it allows seamless switching between simulation and real-world testing, whereby the hardware devices are initialized, parameterized and controlled automatically, too.

Another benefit of the centralized test-management is the possibility of automated execution of pre-defined testcases in both the simulation and real-world tests. Possible parameters of such a testcase are, for example, type and number of nodes, parameterization of each individual node, type of channel (measured/model) and type of Hardware (USRP, daughterboard, generic model).

Besides from that, special GNU Radio applications, called WiNeLo probes, can be attached to the network. These probes replace real-world measurement equipment like spectrum analyzers in the simulation mode. They supply the developer with efficient tools for error detection and debugging

IV. Live Demonstration

The demonstration shows the capabilities of WiNeLo serving as a SDR development environment within the context of a typical development cycle. Considering that, we make use of two basic GNU Radio example applications as well as an newly developed adaptive frequency hopping ad-hoc system. While changing the communication parameters (modulation, noise power, channel model), we compare simple performance metrics such as simulation duration, packet and bit error rates. In addition, new debugging possibilities such as the WiNeLo probes are presented.

Fig. 4: Real-world test setup

The test-management interface and switching between simulation and real-world testing is presented live using the hardware setup seen in **Figure 4**. Thereby, conference attendees are able to change test parameters, check out some of the new WiNeLo probes and evaluate the overall simulation performance (5 - 10 times slower than real-time).

V. Conclusion

New development tools need to be created in order to face the changed development process of wireless systems.

We show that an extended version of WiNeLo is able to serve as a SDR development environment and thus shorten development cycles. Therefore, the existing implementation is extended by basic hardware models (incl. mixing and rate-conversion), accurate modelling of the timing behaviour and a centralized test-management component.

Moreover, the introduced development techniques are shown within the scope of the demonstration.

References

[1] J. Elsner, M. Braun, S. Nagel, K. Nagaraj, and F. K. Jondral, "Wireless Networks In-the-Loop: Software Radio as the Enabler," *Software Defined Radio Forum Technical Conference, Washington DC*, 2009.

[2] "GNU Radio Website." [Online]. Available: www.gnuradio.org

[3] G. Baier, "Channel Modelling and Analysis for the Simulation of Wireless Networks," Master's thesis, Karlsruhe Institute of Technology, 2012.

[4] S. Koslowski, M. Braun, J. P. Elsner, and F. Jondral, "Wireless Networks In-the-Loop: Emulating an RF front-end in GNU Radio," *SDR Forum 2010 European Reconfigurable Radio Technologies Workshop, Mainz, June 25, 2010*, 2010.

[5] "USRP Hardware Driver." [Online]. Available: http://ettus-apps.sourcerepo.com/redmine/ettus/projects/uhd/wiki