Flexicells and SDR4ALL - A cognitive radio test bed

Sylvain Azarian, Mérouane Debbah

FLEXICELLS AND SDR4ALL– A COGNITIVE RADIO TEST BED

Sylvain AZARIAN – Mérouane DEBBAH

Alcatel-Lucent Chair on Flexible Radio, Supélec, France
{sylvain.azarian, merouane.debbah}@supelec.fr

ABSTRACT

Advanced cognitive radio algorithms involve different technical fields like signal processing, wired network communication, computing, etc. Prototyping such systems raise new issues when one wants to find 'off the shelf' bricks, ready for experiments: the integration step is time consuming, is not free of risks of non-compatibility and most of the time does not fall in the scope of a PhD work. For these reasons, many research groups have invested in building a testing platform for their cognitive radio activities. Because the Alcatel Lucent Chair on Flexible Radio is engaged in research covering a wide field of topics, the need for a fully customizable experimental test bed rose and was initiated with SDR4ALL on the radio side, and continued with the ***FlexiCells*** *project for the infrastructure. This paper presents the platform we are designing at Supélec.*

1. INTRODUCTION

Simulation is usually the final step in telecommunication research. The main issue comes when the simulation must be done at the system level: mathematical models can be set for sub-problems but it can become quickly challenging when a full cognitive network has to be evaluated, when the behavioural simulation must be mixed with signal processing simulation. Typical tools like Matlab are not at their best when different levels of the communication stacks must be simulated. At that point, one often wants to setup a simulator, also known as a "simulation platform". Several testing environments have been developed by academic research teams worldwide, targeting different classes of applications (see [1][2] for example). One key feature of any simulation platform is its capability to be used by researchers and PhD students with limited requirement of technical staff manpower, but doing a universal test bed, customizable at all levels is still often a dream, mainly because of cost constraints, or because of some key sub-blocks involved cannot be easily opened or shared with other groups, often because of Intellectual Property issues.

Because of the wide-scope research done by our group, none of the existing platforms could be used "as is". The ***FlexiCells*** project is an initiative to create an evaluation platform for cognitive radio experiments. In this project, the following items are under study:

- Spectrum sensing / allocation,
- Interference management,
- Radio device reconfiguration,
- Backbone packet scheduling.

These topics are part of the "cognitive / flexible radio" research works performed in the Alcatel-Lucent Chair at Supélec [1]. ***FlexiCells*** can be considered as a realistic test environment for cognitive and flexible radio based on Software Defined Radio devices. This type of architecture replaces the protocol/modulation specific hardware by digital signal processing, hence opening 'on the fly' change of radio systems by just reloading appropriate signal processing code [2]. To avoid long design cycles, this system has been developed to be 'research friendly' and tries to hide all low-level details (hardware and software) behind high level languages and functions. For example, all the hardware setup can be made from Matlab [3]. Of course, for power demanding applications, a specific API (Application Programming Interface) has been developed to ease integration in "low level" custom-made applications. The system we designed aims to recreate in a limited environment realistic operating conditions. In this scheme we target the following structure:

- A primary (or legacy) system of communication made of one "base station" and "mobile nodes". Typically we simulate here the case of a phone base station connected to several mobile users.

- A set of cognitive (secondary) users. These users will initiate point to point communications, first by finding the most appropriate frequency and modulation schemes (this is the sensing/allocation part), and then estimate the best scenario to limit interferences with the primary system. This sensing step can be done locally (embedded algorithm) or remotely (spectrum broker). To operate, the mobile system may not have the relevant signal processing code aboard, and may download and upgrade its firmware, this is the reconfiguration part.

- A local area network, to interconnect some of the nodes, as the “backbone”. This network is here to recreate and test the wired infrastructure used in mobile telecommunications, to experiment routing and scheduling algorithms.

Such scenarios have been studied in depth in the team, in the VFDM context [4][5] or in the "small cell" context [6]. ***FlexiCells*** is an attempt to build an experimental platform to benchmark these works. By itself the platform can be seen as a "container" for studied algorithm and does not implement any of them directly, but provides placeholders for inserting in the simulator different types of behaviours and benchmark them.

A set of metrics are created to evaluate the performances of the tested algorithms. For example, the interference created by the “cognitive network” communications on the primary users is one of the key indicators for the cognitive system performance. These indicators are logged in a database for comparison and benchmarking. Figure 1 illustrates the ***Flexicell*** simulation environment.

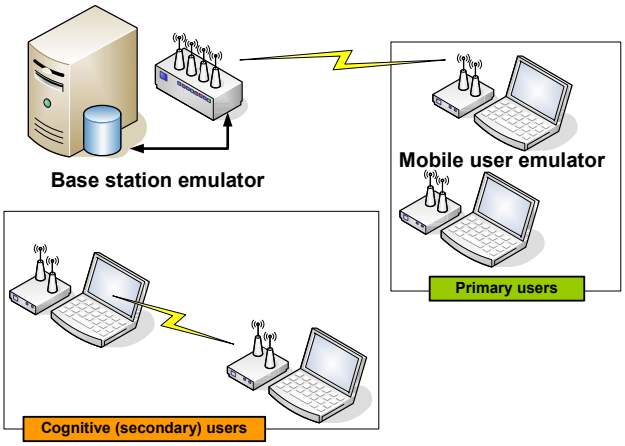


Figure 1 - The FlexiCell testing platform

This system requires specific hardware and of course specific software. Our approach and achievements will be described in more details in the following sections.

2. HARDWARE DESCRIPTION

We are dealing here with different types of requirements, and the challenge is to manage a base-station with multiple user capabilities, base station for single user, spectrum sensing nodes, and mobile user nodes.

2.1 Base station and RF monitoring

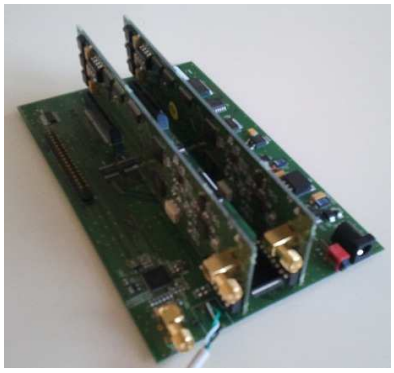
For the base station emulation, a MIMO system has been selected, using a multiple channel system. One Pentek 78620 multichannel board [7] is dedicated to implement a MIMO 2x2 system and one synchronization channel. This board comes with 2 ADC with 200 Mega samples / second and 3 DAC channels with 800 Mega samples /second.

2.2 Mobile terminals and sensing nodes

The mobile users system relies on a SDR4ALL board [8] and a small computer, running the code to emulate a primary or secondary user. Briefly, the SDR4ALL hardware can be described as a simple direct-conversion transceiver, using the USB bus to transfer baseband samples to and from a host PC, hence limiting the achievable bandwidth to some 8 MHz, because of USB2 performances. The SDR4All hardware is made of:

- One main board containing the USB Bridge + logic (FPGA),
- One or two daughter boards for radio transmission (SISO or MIMOx2 communication mode)

The embedded processor processes the USB request and forwards the relevant commands to internal peripherals. This task is managed by the firmware. Initially developed as a standalone radio tool for research and education, SDR4ALL comes with a lot of applications [, some of them have been rewritten to open SDR4ALL integration in the ***FlexiCells*** framework.



The SDR4ALL boards

The sensing nodes are also using the SDR4ALL modules. In this case, a specific firmware is placed in the onboard FPGA to allow wider-band sampling in burst mode. In this case, an embedded FIFO is filled at maximum

sampling rate and transferred to the host PC for spectrum monitoring.

3. SOFTWARE AND NETWORK INFRASTRUCTURE

Three typical scenarios are used in ***FlexiCells***:

- Local baseband processing: the SDR4ALL board samples are processed locally, by the local computer. In this scheme, the digital signal processing code is running on the local PC, and only monitoring data is transferred on the backhaul network.

- Remote baseband processing: for algorithm tuning or interference estimation, any node can stream received RF samples to a remote processing node on the network, to a custom or Matlab application.

- Benchmarking: in many situations, we want to compare the achievements of two different algorithms, running simultaneously on the same data. For example, free spectrum estimation can be done using different methods. It is then useful to run different techniques simultaneously, using the same dataflow from the same sensor nodes and compare the results.

A study of existing network RF streaming systems was conducted and the VITA 49-VRT [9] studied in depth. The VRT protocol describes the network packet structure to transfer samples or commands between equipments. A simplified version of the network packets and protocol has been implemented in the system, but keeping the VRT philosophy.

Redundancy and distributed processing: When more computing power is required, it is often easier to add a computer somewhere on the network and duplicate the data stream instead of reorganizing a running system. This opens loading and off-loading of complex processing without stopping a running monitoring system. Distributing processing also adds more reliability to the system where the same algorithms can be running over multiple processing nodes.

The resulting architecture is a "Software Defined Radio in the Cloud" system where key processing nodes can be distributed over a private or public network. This is described in figure 2.

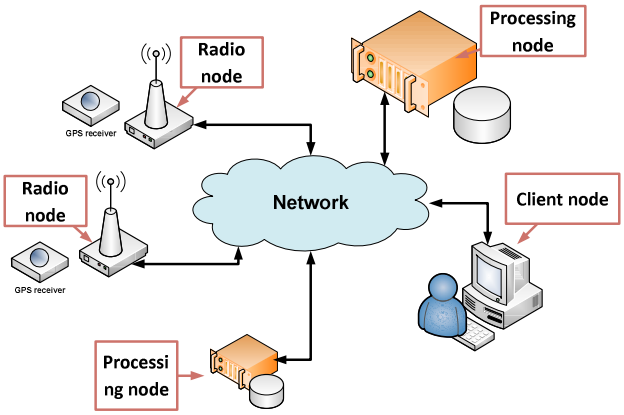


Figure 2 - Software defined radio in the Cloud

The following sections describe the different types of nodes introduced in figure 2.

3.1. Radio nodes (sensors)

A radio node is a software defined radio receiver (using SDR4ALL) controlled by a local processing unit, typically a computer running a Linux operating system. It is connected to the TCP/IP network, using a wired or wireless interface. A radio node can be turned on or off at any time. On startup it opens a TCP/IP connection to a predefined server node to declare its availability. Commands and samples can be transmitted to the relaying node in paquet frames over the network using a VRT-like scheme. One radio node can be connected to more than one relaying node at the same time.

A typical radio node is organized as follows (see figure 3): Control messages are sent to the SDR4ALL device to set the working frequency, sampling rate, and other settings like RF gain, filters etc. This interface is managed through one "hardware abstraction layer" (HAL) specific to the radio device to ease integration of different types of hardware. A local GPS receiver is also connected to the HAL, giving GPS time, position estimation and 1PPS (1 pulse per second) synchronization.

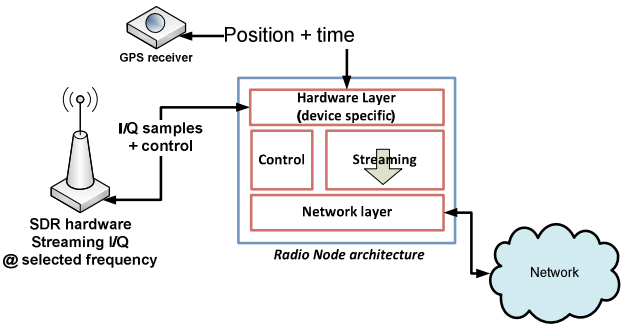


Figure 3 - Typical radio node in FlexiCells

All the network connections are managed through a "network engine" taking care of the connection to the servers: message building and transmitting, message receiving and decoding. The network engine is also organized in two main families of code: one API common

to all the connected applications and one "receiver specific" code (Application code) – see figure 4.

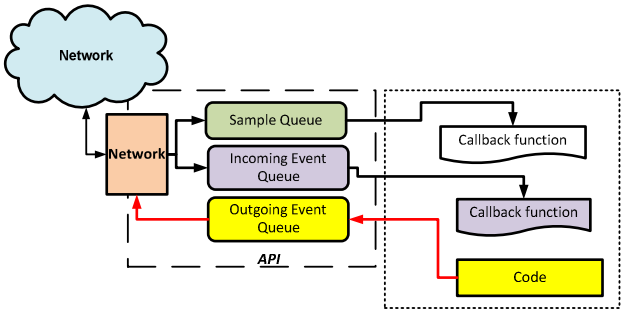


Figure 4 - Client nodes software architecture

Network packets are processed and then stored in priority queues. Depending on message type, some specific events are triggered. For example, when the remote server asks for streaming start or stop, a specific function is called inside the application. This avoids going through all the packets and shortens the latency time of the system.

3.2. Server nodes

These nodes have to

1. Handle remote nodes connection,
2. Handle remote radio streams publishing and synchronization,
3. Manage client node subscription to streams,
4. Dispatch commands,
5. Dispatch information messages.

The heart of the server is organized around a packet scheduler, analysing message queues (priority queues) incoming from remote nodes and dispatching messages (see figure 5).

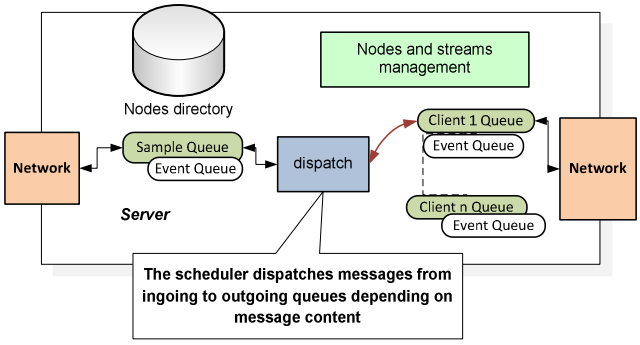


Figure 5 - Server architecture

A typical dispatch case is illustrated in figure 6 where client node A wants to start a remote radio device:

1. At time (1) it sends a ***start*** message to server.
2. This message is forwarded (2) to the radio node.
3. When the node has started, a feedback message ("status information change message", (3) on figure) is sent to the server.

All nodes who subscribed to radio node receive the status update (messages (4) on figure).

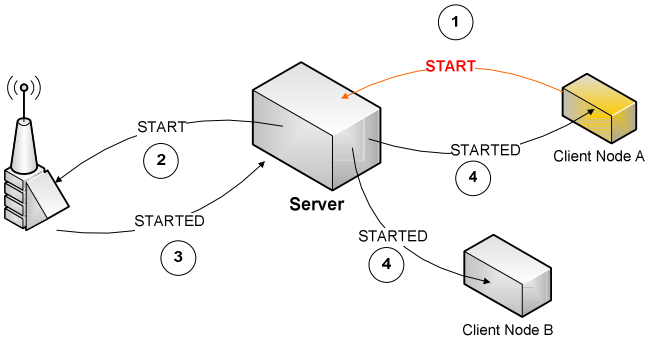


Figure 6 - Dispatching messages across the network

The very same situation happens when one client node wants to change for example the working frequency. The request follows the same path, and acknowledged according to the same scheme.

This scheme avoids client to radio node direct connection but adds some latency for message re-dispatching, but it opens stream sharing between multiple clients, hence permitting multiple use of same real-time RF data by different applications.

3.3. Client nodes

Based on core structure presented here, different applications have been written to test the system:

- Command line application to remotely start / stop streams;
- Matlab Client;
- SDR# client : SDR# is an open-source SDR application providing simple demodulation and waterfall display, initially developed to decode AM, FM and SSB audio on HF bands;
- Custom GUI to control the network nodes and display their location on a map (using Open Street Map cartographic public servers).

4. CURRENT STATUS AND FUTURE WORKS: DISTRIBUTED SYNCHRONIZATION

Platform integration is still in progress and first simple point to point transmissions are now working, controlled remotely from Matlab or dedicated C Api. Next big step is the distribution of synchronization over nodes. In a typical mobile network, the base stations use a GPS synchronized clock and all the time-critical signals are derived from it. In our indoor scenario this is not very realistic.

The issue comes when we want a time-base to synchronize the primary and secondary user communication. One solution under evaluation is to use a specific signal transmitted by the primary base station (see §1.1), working as a "synchronization beacon". A matched-filter is coded in the SDR4ALL FPGA and waits for a predefined code. Once this code is received, the samples are delivered (or transmitted) after a predefined time delay.

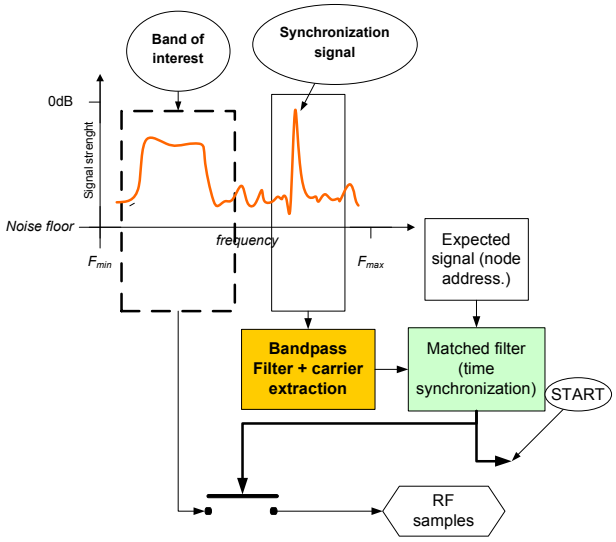


Figure 7- Remote synchronization

The Walsh - Hadamard codes are good candidates as it is easy to generate a set of orthogonal codes with limited cross-correlation, this solution is currently under evaluation.

We also need to decide the different algorithms we will implement for the spectrum management. One first approach will be to sample continuously the whole ISM band and keep the data and compare offline with other possible methods to benchmark them.

5. CONCLUSION

The ***FlexiCells*** platform still requires a important amount of work to be fully running, several issues in the hardware, embedded or infrastructure layers have to be fixed to reach our initial goals. We hope to have soon a completely "research friendly" reconfigurable platform, opening promising fields of experiments.

6. ACKNOWLEDGMENTS

We would like to thank the Digiteo Foundation for its funding, and the french company SOVENTIS for providing us with some key infrastructure tools.

REFERENCES

[1] The FIT experimental facility (http://fit-equipex.fr/testbeds/cognitive-radio).

[2] Cores - A cognitive Radio Testbed by the UCLA (http://cores.ee.ucla.edu/index.php?title=CR_Testbeds)

[3] The Alcatel-Lucent chair on flexible radio : http://www.flexible-radio.com/

[4] J. Mitola III and GQ Maguire Jr. "**Cognitive radio: making software radios more personal**". IEEE personal communications, 6(4):13–18, 1999.

[5] Cardoso, L. S., S. Azarian, P. Jallon, and M. Debbah, "**SDR4all: Software Defined Radio Made Easy**", *6th Karlsruhe Workshop on Software Radio*, Karlsruhe, Germany, 2010.

[6] Maso, M., L. S. Cardoso, E. Baştuğ, N. Linh-Trung, M. Debbah, and O. Ozdemir, "**On the practical implementation of VFDM-based opportunistic systems: issues and challenges**", *REV Journal on Electronics and Communications*, vol. 2, no. 1-2, 2012.

[7] Maso, M., L. S. Cardoso, E. Baştuğ, M. Debbah, and O. Ozdemir, "**VFDM: a prototype of cognitive transceiver**", International Workshop on Communication Systems (IWCS), Hanoi, Vietnam, 2011.

[8] Mawlawi, B., E. Baştuğ, C. Nerguizian, S. Azarian, and M. Debbah, "**Non-Invasive Green Small Cell Network**", 46th Annual Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, California, USA, 2012.

[9] Pentek http://www.pentek.com/

[10] SDR4ALL : http://www.flexible-radio.com/sdr4all

[11] The VITA-49 VRT "VITA Radio Transport protocol" - The VITA Standards Organization