SOFTWARE ARCHITECTURE DESIGN FOR A DYNAMIC SPECTRUM ALLOCATION-ENABLED COGNITIVE RADIO TESTBED

Oscar Tonelli, Gilberto Berardinelli, Andrea F. Cattoni, Troels B. Sørensen and Preben Mogensen

Department Of Electronic Systems, Aalborg University
Niels Jernes Vej 12, 9220, Aalborg Ø, Denmark
email: ot@es.aau.dk

ABSTRACT

The evolution of wireless communications is bringing into reality the dense deployment of femto and local area cells, which represents a challenging scenario for proving the effectiveness of Cognitive Radio (CR) frameworks. In particular the Dynamic Spectrum Allocation (DSA) paradigm aims at solving the resource allocation problem in a fully autonomous way. While in both CR and standardization lot of effort has been spent in developing efficient DSA algorithms, their research-oriented software radio implementation is still disregarded. In this paper, the design approach used for the definition of a CR-based, local area-oriented testbed platform is introduced. In particular, the testbed design in relation to the selected scenario and the architecture of the developed software framework are presented. An analysis of the performance of key software functionalities is also provided: preliminary results show a computational load compatible with nowadays Commercial-Off-The-Shelf computers.

1. INTRODUCTION

The wireless communication standards of future generation (e.g. International Mobile Telecommunications-Advanced, IMT-A [1]) aim at increasing the transfer data rate at the user end. In order to achieve this goal, advanced techniques such as multiple antennas transmission are adopted and a wide channel bandwidth is required. In addition, standards such as the Long Term Evolution - Advanced (LTE-A), consider the deployment of femtocells in local area as a necessary solution to boost the network capacity. The assignment of wide frequency bandwidths, may lead to operative communication scenarios where the operators are forced to share the same portion of spectrum; the possible uncoordinated deployment of femtocells is an additional factor that is expected to increase the complexity of the interference scenario. The upcoming challenge is therefore to provide solutions for the dynamic sharing of frequency resources among interfering nodes in the network.

The Cognitive Radio (CR) [2] concept features the acquisition of knowledge about the surrounding environment in order to optimize and reconfigure the main communication parameters. There is great interest in applying the Cognitive Radio concept to the development of Dynamic Spectrum Allocation (DSA) algorithms, which enable an autonomous allocation process of channel resources on a cognitive device. Example of application of such concepts can be found in [3], where an approach based on Game Theory is applied to the distributed spectrum sharing problem.

Other recent contributions in literature, faced the problem of interference management in a spectrally overcrowded deployment: validation studies of these theoretical proposals typically rely on Monte Carlo simulations. A challenging research area is then related to the real-world implementation of CR theoretical concepts: Software Defined Radio (SDR) is considered the enabling technology for CR design given the possibility of implementing systems that are completely configurable in software. A popular example of SDR framework is GNU Radio [4], a component-based development toolkit, particularly suited for the realization of Physical (PHY) layer transceiver chains.

Since the practical implementation of CR concepts on real-world platforms, is often affected by several software and hardware limitations, the efficient design of SDR-based testbeds is currently a topic under investigation.

Most of current work on CR SDR-based testbeds addresses the problem of opportunistic usage of the available spectrum; specific topics such as the exploitation of TV vacant channels and the creation of ad-hoc networks are of major interest. Great effort has also been spent in the design and implementation of high-performance and standard-inspired PHY architectures. OpenAirInterface [5] for example, is a software-based platform aiming to support the LTE air-interface, Iris [6] instead, is another software radio platform that features runtime reconfigurability and support for multiple hardware solutions.

The goal of this paper is to present the authors' approach in the design of a SDR-based testbed able to provide validation and experimentation support for CR-based algorithms. Particular attention is given to the local area/femtocell deployment scenario and to the coordination issues among nodes in the network. Existing SDR system such as GNU Radio, provide support mostly limited to the PHY layer, other optimized platforms have limited reconfigurability capabilities while Iris code is not yet available to external developers. All these considerations lead to the development of an indipendent solution. The possibility of developing real world implementations of CR algorithms over the testbed platform, significantly contributes to their Proof-Of-Concept and further optimization. An easily reconfigurable and deployable SDR-based testbed moreover, represents a practical solution for local area/femtocell scenarios.

The paper is structured as follows: Section II introduces the features of the testbed platform to be implemented in relation to the selected scenario, Section III illustrates the architecture of the developed software framework; Section IV describes how the technical challenges given by the required features are addressed in the system design, Section V discusses the software performance providing computation results. Section VI finally provides comments and conclusions with future work plans.

2. TESTBED DEFINITION

The design of the testbed aims at providing adequate support for the experimentation of cognitive principles in femtocell deployment. The work carried on in the project SAMURAI [7] provided initial inspiration and scenario references; in particular the scenario individuated in [8] is considered. In this paper the authors propose an algorithm for the autonomous management of interference among the nodes in a LTE-A network. The main features of the testbed, are the support for multiple Component Carriers (CCs), an Orthogonal Frequency Division Multiplexing (OFDM)-based air-interface and Time Division Duplexing (TDD) connection among network entities. The goal of the testbed is to prove the effectiveness of the selected DSA algorithms in a challenging scenario, therefore a situation of overcrowded spectrum is considered as a basic design requirement. The testbed network is composed by a number of nodes acting as Access Points (APs) or femtocell Base Stations (BSs) and an equal number of associated User Equipment (UEs) devices; each AP is able to communicate with the other APs in the network through a Cognitive Pilot Channel (CPC) [9]. Figure 1 shows the connections between the testbed devices.

The available spectrum in the system is divided in a number of CCs that the APs may allocate for the communication with the UEs. The number of APs in the network is chosen accordingly to the number of CCs, in order to determine a resource competition scenario among the APs. The allocation process of the CCs is triggered by the AP traffic requirements and is performed accordingly to the DSA algorithm executing on the network nodes. The TDD link provides the connection between the AP and the UE, and can also be exploited in the cognitive process for sensing purposes: measurements collection from the UE can provide the AP information about the link channel quality and the presence of interferers, thus impacting the CC allocation decision. The CPC moreover, is used for exchanging informations among the APs, thus enabling the final decision on the spectrum allocation.

The cognitive femtocell scenario exposes two main design challenges to the practical realization on an SDR-based testbed. A first design consideration is given by the necessity of carrying information data over channels which are characterized by a multiple CCs structure: the Radio Frequency (RF) front end is required to support a suitable frequency bandwidth and the OFDM-based PHY layer should be able to manage it properly. Secondly, the testbed aims at providing a CPC that may not benefit the support of a backhaul link: the implementation of an Over-The-Air-Communication (OTAC) control channel is a topic that raised interest also in the LTE-A standardization process [10].

3. SOFTWARE ARCHITECTURE

An SDR-based implementation well suits the previously defined testbed scenario: a typical SDR configuration features an RF-capable device which can be completely configured and controlled via software by an host PC, thus meeting the flexibility requirements of a practical experimentation. The proper realization of an SDR-based testbed is highly dependent to the selected software architecture.

In this paper the developed architecture takes into account the following concepts:

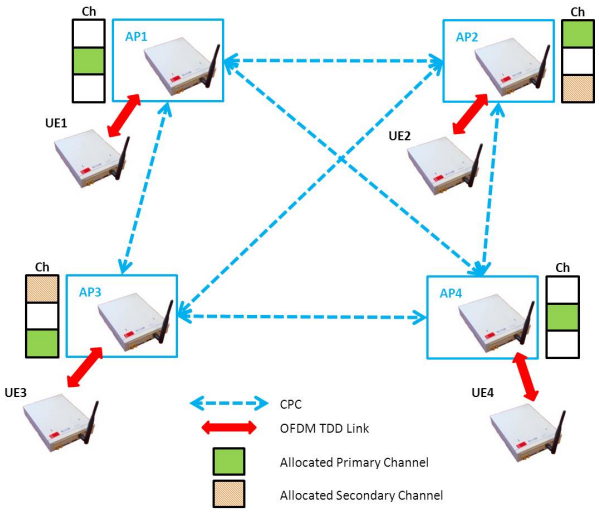

Figure 1: Testbed Network.

- Testbed purposes: experimentation and optimization activities require complete access to system parameters at every level of the protocol stack.
- Real-world context: the software architecture must be able to handle multiple parallel tasks which may run asynchronously.
- Runtime reconfiguration of parameters: allows to handle dynamic testbed scenarios.
- Platform development: support for the development of additional testbed functionalities.

In order to achieve the desired level of parallelism and reconfigurability that a real-world development platform requires, the implemented software features an object-oriented, modular, multithreaded architecture. The main idea is to develop specific tasks in independent functional modules, and group coherent modules in more complex subsystems: these components can be then assembled in order to obtain the complete set of functionalities of a communication node. A proper management of shared memory, implementation of inter-module interfaces and organization of threads of execution, are the fundamental activities enabling the software concept realization. Figure 2 depicts the software architecture, highlighting the modular implementation; a PHY communication chain, Medium-Access-Control (MAC) functionalities and the Cognitive Algorithm (CA) are implemented as indipendent subsystems. The connections between system components are organized according to a hyerarchical structure, as depicted in Figure 3; every module features input/output interfaces that provide the communication means with other modules; each subsystem is able to manage the interfaces of its inner modules, establishing connections with other subsystems. The inter-module interfaces are designed following specific software patterns, as for example publish/subscribe [11], which target a component-based realization of the system.

The hierarchical architecture of the system also applies to the thread generation and execution: a first-level process generates the main threads of execution for each subsystem, which then internally manages its module-specific threads. The aim is to have a centralized management of the system during runtime, also enabling reconfiguration.

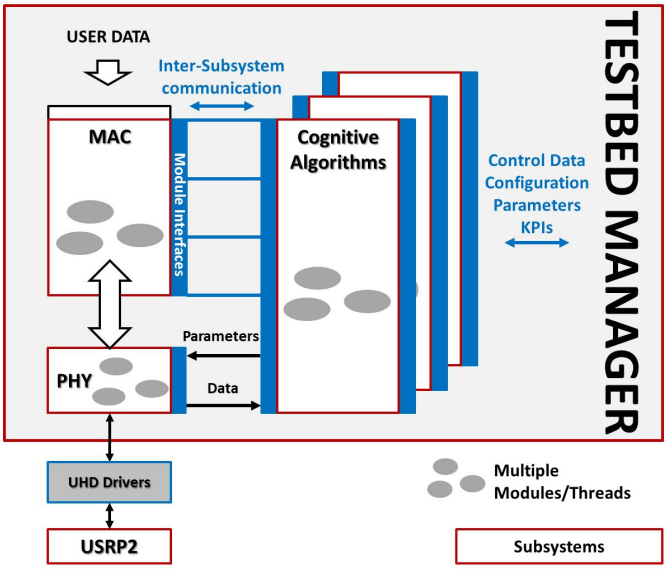

Figure 2: Architecture of the testbed software framework

In order to better support a modular and multithreaded architecture, all the software components have been written in object-oriented C++ programming language. The choice of C++ is due to several reasons: suitability of the code for real-time processing purposes in comparison to other object-oriented languages, availability of libraries for multithreading support, compatibility with the drivers required to interface with the hardware and the possibility of adopting a Test-Driven-Development methodology which exploits the modularity of the code for advanced software testing purposes. C++ libraries also allow easy interfacing with Linux OS system, thus enabling a complete control over the testbed from user-space applications.

Parallel threads execution requires proper synchronization and specific solutions for data exchange. Given the proposed component-based and modular design, each module can potentially execute independently: in order to ensure proper data exchange, synchronization elements such as mutexes and events, have been embedded into the interface objects shared by the modules. The software exploits the optimized Poco C++ libraries [12] for thread synchronization and memory management purposes. The interface with the SDR hardware is enabled by the usage of the Universal Hardware Driver (UHD), recently developed by Ettus Research [13] and mainly conceived to operate with the Universal Software Radio Peripheral (USRP) series boards also developed by Ettus. The UHD exploits an UDP-based connection to send and retrieve data from the hardware. A very interesting feature of the UHD is the metadata payload that is attached to the data transferred between the board and the host computer: the metadata field allows to specify time-related settings which provide precise information on the transmission and reception time of the samples. This feature is foreseen to enable greater flexibility for the data flow control also when operating in the OS user space.

One of the greatest concerns related to the software realization of a wireless transceiver, is the computational burden of the implemented features. In particular, the execution of

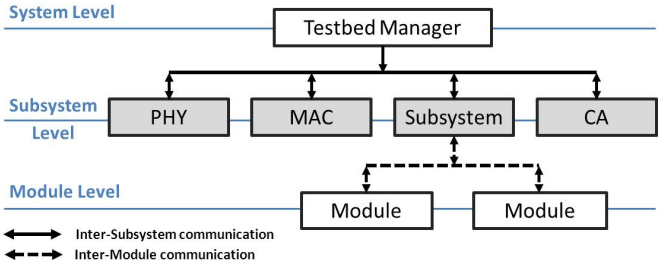

Figure 3: Components Hierarchy and Communication

PHY layer tasks, such as (I)FFT and symbol/frame synchronization routines, as user-space processes, is significantly resource consuming and poses severe limitations to the real-time execution of the code. High data rate transmission is however not the major focus in a testbed conceived for the experimentation with DSA algorithms; the main goal is instead to provide network-wide reconfigurability and ensure a proper interference scenario among the nodes. The software design choice has been therefore to favor a more flexible and easily reconfigurable user-space implementation, rather than boost the real-time features which may require to run tasks in the OS kernel space or directly on the hardware [15].

Latest generation General Purpose Processors (GPPs) on host computers feature multiple cores and threading techniques able to dynamically share the computation load; a multithreaded application in comparison to single-threaded can greatly exploit this advanced hardware features, thus achieving increased performance. Modularization of the code aims also at improving the setting of threading priorities according to the functional task: optimal priority management is required in order to regulate the execution of a high number of threads on the GPP.

4. SYSTEM DESIGN

The developed testbed relies on SDR-based systems for the implementation of the network nodes. The main requirements of the SDR hardware involve flexibility of the RF front-end, software configurability, compatibility with the host and a local area deployment. Ettus USRP version 2 (USRP2) is one of the most popular SDR platform used for CR implementation purposes and seems to fulfill all the requirements. All the nodes in the testbed exploit USRP2 boards equipped with Ettus XCVR 2450 daughterboards as RF front end.

The USRP2 choice allows the deployment of a high number of nodes thanks to the compact hardware solution and affordable costs; the featured Gigabit Ethernet connection ensures great compatibility with modern host-computers and enables flexible testbed deployment in indoor area as proved in the Virginia Tech CORNET testbed [14]. The compatibility with a wide set of daughterboards with different RF capabilities, ensures further potential improvements of the testbed.

The first testbed design challenge is the definition of transmission channel parameters and CCs structure: practical experimentations with the hardware suggested to define 3 CCs in the available RF channel bandwidth. Assuming that a single (I)FFT process is used for generating/retrieving the time samples, it is possible to access one or more CCs

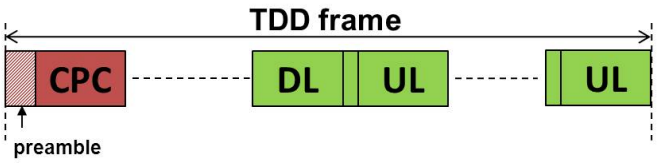

Figure 4: Testbed TDD framing structure

by simply allocating the data over a portion of the (I)FFT bins. The choice of OFDM as air interface is mainly driven by two considerations: the modern wireless communication standards inspiring the testbed (e.g. LTE-A), feature OFDM given its flexibility as well as its multipath mitigation capabilities; in addition, the OFDM frequency granularity may allow the coexistence of information related to different nodes on the same CC. For example node-specific pilot patterns can be implemented and exploited for measurements and node identification purposes.

Another critical aspect in the testbed realization is the design of the CPC. The aim is to support an OTAC control channel enabling the execution of coordinated algorithms which do not rely on a wired backhaul link. In order to fit the TDD frame structure, the control data are time multiplexed in pre-allocated CPC frames; the adopted solution is depicted in Figure 4, which shows the CPC slot included in the TDD frame featuring Downlink (DL) and Uplink (UL) slots. This OTAC implementation implies that the APs must achieve an inter-AP synchronization in addition to the AP-to-UE synchronization. A practical solution to the problem is the definition of a global time reference provided by a master node in the network; the master node generates a frame preamble which is known by all the nodes in the network and therefore can be used as a reference for the frame synchronization. Another option for the transmission over the CPC is to allow unsynchronized access by implementing a slotted ALOHA protocol.

All the design solutions such as (I)FFT based transmission, TDD framing and system synchronization, have been implemented as specific software tasks and integrated in properly defined functional modules according to the proposed software architecture. The concurrent execution of multiple modules comes at the cost of computational tasks that need to run with a high degree of parallelism and strict latency constraints. The performance of the designed system is therefore highly dependent on the characteristic of the host General Purpose Processor (GPP) and on how the software modules exploit the available resources.

5. SOFTWARE ANALYSIS

In order to ensure the proper execution of the implemented solutions, it is extremely important to analyze the behavior of the software modules during the execution on the host computer. In particular, an evaluation of the computational load on multi-core machines allows to individuate possible optimization areas. Computational load tests have been performed on the testbed host computers which feature Ubuntu 10.10 OS and multi-core processors. The host computers considered are equipped with Intel Core i7 740Q 1,73GHz and Intel CoreDuo Quad Q9100 2,26GHz processors; even though both processor are quad-cores, the Intel i7 features

Table 1: PHY parameters

Frame Size	50 TTI + Preamble
FFT-size	256
Cyclic Prefix length	48
Nr. Time Symbols	13
Tx Rate	10MS/s
Modulation	QPSK
Coding	Convolutional 1/3

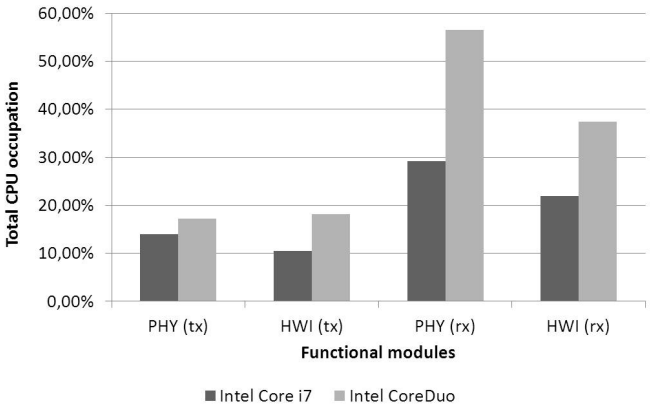

Figure 5: CPU resources occupation analysis

the Intel ®Hyper-Threading Technology which allows to operate over 8 logic cores, enhancing the parallelism opportunities.

The selected software functionalities for the analysis, are related to the PHY layer tasks at the transmitter and receiver; the initial Primary CC selection presented in [8] has also been analyzed in order to obtain an estimate of the computational impact of basic cognitive tasks. The results for each functional module, in terms of total CPU resources occupation percentage, are presented in Figure 5.

The PHY transmitter (tx) features data generation, coding and modulation according to the parameters in Table 1; the same communication parameters are used at the PHY (rx), which instead includes synchronization, detection and de-modulation/decoding tasks. The PHY layer processing has been disjointly analyzed from the hardware interfacing tasks (HWI (tx and rx)) which are mainly related to the transfer of the data samples between the host and the USRP2 boards. The implemented cognitive algorithm features interference measurements at the nodes and exchange of information about CCs allocation, on the control channel. The cognitive decision process is based on such data and proven to have a negligible computational burden for a low number of nodes in the network; numerical results have therefore not been included in the histogram.

Figure 6 provides results in terms of average concurrency of the modules threads on the processors. A value of 2 means that during the execution time, the is an average value of 2 threads concurrently occupying resources on the processor's cores.

The obtained results provide several design indications for the testbed: firstly, the considered CR algorithmic func-

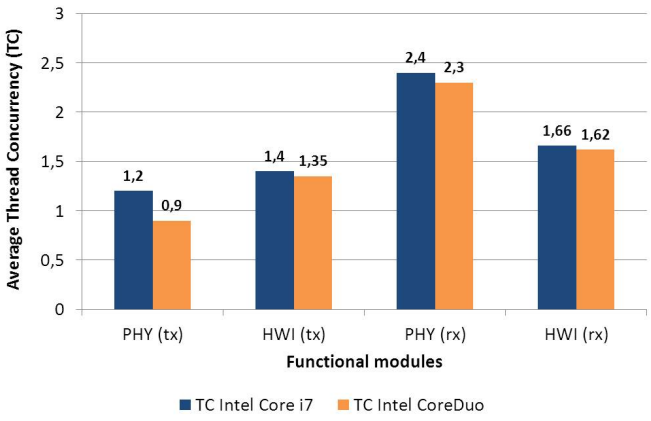

Figure 6: Average thread concurrency during execution

tionalities do not pose severe computation constraints, while the PHY layer tasks impact is considerably heavier, with a relevant computational burden especially at the receiver side. The resource occupation on the Core Duo is more critical than on the Core i7, mainly due to the fact that the Hyper-Threading technology allows to almost double the available resources with the same amount of physical cores. Although in a TDD system the TX and RX tasks are most of the time not concurrent, the system execution on the Core Duo hosts requires further optimizations. The threads concurrency analysis finally, shows that the implemented modules do not fully exploit the parallelism capabilities of the processor; the target thread concurrency should be indeed close to the number of logic cores (e.g. 4 or 8). Although the Core i7 hosts provide satisfactory performance with the PHY layer and the cognitive algorithm, the execution of the system also needs to consider the implementation of MAC layer protocols that produce additional load on the host GPP.

6. CONCLUSIONS

In this paper a design approach for the realization of a CR testbed platform has been proposed; the main goal is the experimentation with DSA algorithms in femtocell deployment scenarios. The development of a modular and multithreaded software framework has been individuated as a suitable solution for achieving the testbed goals in terms of flexibility and reconfigurability of the platform, in the selected operative context. Preliminary computational load results presented in this paper, allow to identify the PHY receiver as one of the most critic system functionalities and highlight the greater suitability for the system of processors featuring high level multi-threading capabilities such as Intel Core i7. Thread Concurrency results suggest to improve the threading management in order to fully exploit the computational resources. Future work will also include the implementation of additional testbed modules able to support increased complexity at MAC layer and feasibility studies of system execution on dual-core processor with Hyper-Threading technology.

ACKNOWLEDGMENTS

The work described in this paper was carried out with the support of the SAMURAI-project ("Spectrum Aggregation and Multi-User MIMO: Real-world Impact"), a collaborative project funded by the European Commission through the 7th ICT Framework Programme.

REFERENCES

[1] Requirements Related to Technical Performance for IMT-Advanced Radio Interface(s). ITU-R, Rep.M2134.

[2] S.Haykin, "Cognitive Radio: Brain-empowered wireless communications," *IEEE Journal on Selected Areas in Communications*, vol.23, no.2, pp.201-220, February 2005.

[3] J.E. Suris, L.A. Dasilva, Z.Han and A.B. Mackenzie, "Cooperative game theory for distributed spectrum sharing, " in *Proc. IEEE ICC*,pp. 5282-5287, Glasgow, June 24-28. 2007.

[4] GNU Radio, http://www.gnuradio.org/.

[5] F. Kaltenberger, R. Ghaffar, Raymond Knopp, H. Anouar, C. Bonnet, "Design and implementation of a single-frequency mesh network using OpenAirInterface," *EURASIP Journal on Wireless Communications and Networking*, Article ID 719523, 2010.

[6] P. D. Sutton, J. Lotze, H. Lahlou, S. A. Fahmy, K. E. Nolan, Bariş Özgül, T. W. Rondeau, J. Noguera, L. E. Doyle, "Iris: An Architecture for Cognitive Radio Networking Testbeds," *IEEE Communications Magazine*, vol.48, no.9, pp.114-122, September 2010.

[7] SAMURAI - Spectrum Aggregation and Multi-User MIMO: Real-World Impact, http://www.ict-samurai.eu/.

[8] L. G. U. Garcia, K. I. Pedersen and P. Mogensen, "Autonomous Component Carrier Selection: Interference Management in Local Area Environments for LTE-Advanced," *IEEE Communications Magazine*, vol.47, no.9, pp.110-116, September 2009.

[9] P. Cordier, P. Houze, S.B. Jemaa, O. Simon, D. Bourse, D. Grandblaise, K. Moessner, J. Luo, C. Kloeck, K. Tsagkaris, R. Agust, N. Olaziregi, Z. Boufidis, E. Buracchini, P. Goria, A. Trogolo, "E2R Cognitive Pilot Channel concept," in *IST Summit*, Mykonos, June 2006.

[10] 3GPP Tdoc R1-090236, "Inter eNB over-the-air communication (OTAC) for LTE-Advanced," Nokia Siemens Networks, Nokia, January 2009.

[11] F. Buschmann, K. Henney, D. C. Schmidt, *Pattern-Oriented Software Architecture: A Pattern Language for Distributed Computing, 4th Volume*. John Wiley Sons, 2007.

[12] Poco C++ libraries, http://pocoproject.org/.

[13] Ettus Research, http://www.ettus.com.

[14] T. R. Newman, S. M. S. Hasan, D. DePoy, T. Bose and J. H. Reed, "Designing and Deploying a Building-Wide Cognitive Radio Network Testbed," *IEEE Communications Magazine*, vol.48, no.9, pp.106-112, September 2010.

[15] S. Mellers, B. Richards, H. So, S. M. Mishra, K. Camera, P. A. Subrahmanyam, R. W. Brodersen, "Radio Testbeds using BEE2," in *Conference Record of the Forty-First Asilomar Conference on Signals, Systems and Computers*, ACSSC 2007, Pacific Grove, USA, November 4-7. 2007.