A Unified Solution to Cognitive Radio Programming, Test and Evaluation for Tactical Communications

Yalin Sagduyu, Sohraab Soltani, Tugba Erpek, Yi Shi and Jason Li
Intelligent Automation, Inc., Rockville, MD, USA
Email: {ysagduyu, ssoltani, terpek, yshi, jli}@i-a-i.com

Abstract—Spectrum is limited but the demand for it is growing steadily with new users, applications and services, both in commercial and tactical communications. The current paradigm of static spectrum allocation cannot satisfy this demand, resulting in a congested, contested environment with poor spectrum efficiency. Tactical radios need to share spectrum with other in- and out-of-network tactical and commercial radios, subject to potential jamming and other security attacks. Cognitive radio provides tactical communications with new means of spectrum sharing, cohabitation configurability and adaptation to improve communication rates, connectivity, robustness and situational awareness, all translated to network-centric mission success. A systematic solution to software-defined radio (SDR) programming, test and evaluation (T&E) is needed to address spectrum challenges with the network-centric mission success set as the primary goal. Once equipped with cognitive radio capabilities, tactical radios can quickly and reliably discover the white-space and effectively use spectrum opportunities across time, space, and frequency. This article presents the design and implementation of a visual and modular radio software programming tool that supports easy, fast and radio-agnostic development of cognitive radio and network protocols and security mechanisms. This tool is fully integrated with a unified T&E framework that applies the same SDR solution to high fidelity simulation and emulation tests under a common, controllable and repeatable scenario. This unified approach makes prototyping of cognitive radio capabilities with tactical radios faster, easier and cost-effective.

Index Terms—spectrum; cognitive radio; software-defined radio; tactical radio; tactical network; network protocols; security; radio programming; test and evaluation.

I. INTRODUCTION

Spectrum resources are scarce and they are not efficiently utilized with a static and dedicated allocation of frequency bands and their sporadic use. Tactical communications are typically assigned legacy frequency bands for exclusive spectrum use. However, spectrum becomes crowded with a growing demand (e.g., multimedia) of commercial communications, e.g., LTE, WiFi and unlicensed—national information infrastructure (U-NII) devices. Therefore, new means are necessary to enable tactical radios to share spectrum with commercial users (e.g., mobile and fixed wireless broadband use, LTE and Electronic News Gathering). For instance, AWS-3 auction aims to release 1695-1710 MHz, 1755-1780 MHz and 2155-2180 MHz frequency bands, raising both opportunities and challenges for tactical and commercial users to share the spectrum.

Tactical networks span a significant part of available spectrum resources and should be designed, implemented, tested and evaluated with the goal of improving spectrum efficiency and mission success. The legacy form of a static spectrum allocation poses a major obstacle for an efficient use of limited wireless resources. This challenge has promoted substantial research and development into cognitive radio and software-defined radio (SDR) technologies with network-level perception, learning, adaptation, and optimization for an efficient spectrum utilization. Cognitive radio emerges as the enabling technology to support configurability of radios on-the-fly and access to broader pools of spectrum and more efficient utilization of current wireless resources, playing a key role for the next generation of both commercial and tactical communications [1].

To build the next-generation of tactical network communication systems with cognitive radios, a systematic approach including a full scope of design, implementation, and test and evaluation environment is necessary with the systematic and unified execution of the following five components (shown in Figure 1 (a)).

1. Network protocols: Cognitive radio uses a “cognitive engine” to learn spectrum accurately with small overhead, and then promptly adapts to spectrum dynamics, including channel occupancy, interference, congestion and mobility. Instead of a separated and static protocol abstraction, cross-layer design appears a viable way to integrate spectrum sensing and dynamic spectrum access, routing and other network protocols in a unified cognitive network protocol stack. This protocol stack includes PHY, MAC and higher layers that optimally strikes a balance between performance and overhead/complexity tradeoffs.

2. Network security: The configuration and adaptation of cognitive radio is vulnerable to various forms of attacks due to open broadcast nature of wireless medium in Anti-Access/Area-Denial (A2AD) environments. These attacks range from jamming to insider threats including spectrum sensing data falsification (SSDF), primary user (PU) emulation and protocol-violation attacks. Novel means are sought to combat these security issues and maintain the mission-critical performance in tactical networks of cognitive radios.
a) Context-aware systematic approach for tactical communications with cognitive radios. (b) Scenarios and performance metrics for tactical networks.

3. **Platforms**: The use of commercial radios is likely the first step to quickly prototype, test and evaluate cognitive radio capabilities. The next step is to implement these capabilities with tactical radios. Once mature, cognitive radio technologies can be ported to tactical SDRs with different levels of SWaP-C (Size, Weight, and Power-Cost) characteristics and flexibility to enable cognitive radio capabilities.

4. **Programming**: Cognitive radios require fast and efficient SDR programming of protocol modules. Various commercial tools (such as GNU Radio, REDHAWK, LabView and MATLAB) are available to program and control SDRs at software or FPGA level for general signal processing purposes typically focused on the physical (PHY) layer. In addition, higher-layer protocols are needed as part of SDR programming. For fast implementation, there is a growing demand for a user-friendly visual tool that can automatically generate radio-agnostic codes for full protocol network stack based on specified modules in the SDR architecture.

5. **Test and evaluation**: A cognitive radio involves various tunable parameters to be extensively tested for seamless integration of network protocols. The standard procedure is to separate simulations studies (such as CORE and EMANE) and emulation tests with radios. A systematic T&E approach is needed to unify them under a common, controllable and repeatable scenario: a) scalable and high fidelity simulations, and b) in-lab radio-in-the-loop emulation tests with controlled RF characteristics.

The goal is to improve network-centric mission success according to various criteria including but not limited to assured connectivity, coexistence (cohabitation), spectrum situational awareness, minimum latency, maximum geographic, bandwidth and time availability of spectrum resources and cyber protection. Potential scenarios and performance metrics are shown in Figure 1 (b). To achieve this goal, this article presents the novel design and implementation of visual and modular radio software programming tool to support easy, fast and radio-agnostic development of cognitive network protocols and security mechanisms. This fully developed capability is integrated with a unified T&E environment that applies the same SDR solution to high fidelity simulation and emulation tests under the common scenario control. The novelty of this unified solution consists of two main components:

1. **EZPro**: A visual and modular radio software programming tool is presented to generate a workflow of cognitive radio protocols and security mechanisms, develop individual modules for cognitive radio functionalities, and automatically generate the end-to-end software code that can be used with simulation and emulations tests.

2. **Network control environment**: A unified network control environment is presented to set up nodes, configure them as virtual (for simulation tests) or real (for emulation tests), set up network scenario (including topology, channel and mobility effects), assign the SDR code to virtual nodes in simulations, or port the SDR code to SDR platforms for emulations tests.

   a. **Simulation environment**: High fidelity simulation is based on Common Open Research Emulator (CORE) and Extendable Mobile Ad-hoc Network Emulator (EMANE). CORE provides the high fidelity simulation environment with real network stack and traffic for emulating networks on one or more machines [1]. PHY/MAC is emulated by various models (e.g., RF pipe or 802.11) of EMANE [13].

   b. **Network channel emulator**: The same SDR code used for simulations can be used for emulations tests with real SDR platforms. Network channel emulators are needed to emulate wireless channels...
among SDRs in in-lab tests and support controllable, repeatable and scalable hardware-in-theloop experiments. In the featured setup, network channel emulator (a radio channel emulator supporting a full mesh network connectivity), RFnest [3], is used to emulate radio channels and RF interactions among radios, and support potential interaction of real (emulated) and virtual (simulated) radios. RFnest adjusts channel conditions (such as attenuation, delay, multi-path and Doppler) according to the scenario generated for tactical network communications.

With this unified SDR programming and T&E environment, a user can easily, quickly and reliably design and implement cognitive radio protocols across the network protocol stack, and test and evaluate them with both simulation and in-lab hardware-in-the-loop emulation tests under realistic tactical network scenarios.

The rest of the article is organized as follows. Section III presents a novel systematic solution to network-centric SDR programming. Section IV describes SDR modules for cognitive radio network protocols and cognitive radio network security. Section V presents a systematic unified solution to cognitive radio network test and evaluation. Section VIII concludes the paper.

II. A NOVEL SYSTEMATIC SOLUTION TO NETWORK-CENTRIC SDR PROGRAMMING

A. State-of-the-Art SDR Programming Tools

There are various ways to program cognitive radios. Some examples of the state-of-the-art SDR programming software are listed below:

- **GNU Radio**: GNU Radio [4] is a free and open-source software development toolkit that provides signal processing blocks to implement software radios. GNU Radio can be used with existing SDRs such as Universal Software Radio Peripherals (USRPs), or without hardware in a simulation-like environment.

- **REDHAWK**: REDHAWK [5] is an SDR framework to support the real-time software radio applications. REDHAWK develops and tests software modules called “Components” and composition of Components into “Waveform Applications” deployed on a single or multi-network-enabled computers.

- **LabView**: LabVIEW Communications [6] from National Instruments offers a graphical design environment integrated with the USRP. LabVIEW offers parallel constructs spanning multiple cores, threads and targets, and supports real-time embedded processors and FPGAs.

- **MATLAB**: MATLAB® and Simulink® [7] are used for wireless design, simulation, and analysis on software-defined radio such as setting up SDR hardware with preconfigured radio functions, performing real-time signal analysis and measurement. MATLAB and Simulink provides support packages for popular SDR hardware such as USRP, Zynq and RTL-SDR Radio.

These existing SDR programming tools are typically focused on digital signal processing (DSP) at the PHY layer. The extended capabilities needed to implement cognitive radio solutions effectively with tactical radios include:

- Full network protocol stack (beyond PHY/DSP);
- Executable, radio agnostic code;
- Same code used for simulation and hardware testing;
- Operating system independent;
- Dynamic modular workflow design;
- User-friendly GUI with visual programming tools;
- Modular I/O with message passing;
- Visual module manager with edit/merge functions;
- FPGA support.

B. EZPro as a Visual and Modular SDR Programming Tool

To address these gaps, this article presents the design and implementation of EZPro as a visual and modular “Easy Programming” environment to generate software. The user of this software either leverages the built-in code blocks or generates new code blocks. A user designs a workflow by dragging different blocks into the design panel and connecting tests and the test software in the same environment. EZPro then generates a standalone software from the workflow that is portable and executable outside of the EZPro environment.

A multithreaded backend Python class with input/output ports and input arguments supports each block. Different blocks communicate with each other through their connections over a queueing system that supports multi-threaded and multi-processor communication. EZPro provides tools to generate software for various network layers. Any existing software; i.e. GNU radio software blocks, can also be wrapped as an EZPro block and re-used in the workflow.

Figure 2 shows an example EZPro workflow for the SDR software that will run at each cognitive radio. This workflow consists of several blocks for different cognitive radio functionalities:

- **Spectrum Sensing** block senses the environment and feeds the spectrum occupancy results to the **Attack Countermeasures** and DSA blocks. Additionally, a **Spectrum Database** block can also be implemented that connects to a database and pulls the expected spectrum occupancy results.

- **DSA** block evaluates the incoming information regarding the spectrum occupancy and determines the operating frequency.

- **Routing** block decides on the next hop for the packets.

- **Attack Countermeasures** block analyzes the spectrum sensing results and decides what mitigation techniques will be used to ensure continuous operation.

The user can visually revise/extend the workflow.
Figure 2. (a) Example of EZPro workflow for cognitive radio software, (b) Closer look for an EZPro block with its connections to other blocks.

The four main steps of EZPro (shown in Figure 3) are:

1. Drag an individual code block from existing blocks or design an individual block to make the workflow (EZPro provides designated spots in the template to insert the user code).
2. Connect blocks in the workflow by using Tip tool (there is a separate enumerated connection for each parameter passed among blocks).
3. Configure each block using Parameter Panel.
4. Automatically generate source code for the workflow (use EZPro interactive tools to run and debug program source code) and execute the program.

After these four steps, program source code is automatically converted to a standalone software that is deployable to any systems with general purpose processor (GPP).

The computational cost of running EZPro software alone is negligible (about 0.3% CPU and 2.9% memory usage). The computational cost of the developed software depends on the implementation complexity such as the number of threads and the number of blocks used.

EZPro aims to make the programming of tactical radio and prototyping cognitive radio capabilities faster, easier, and cost-effective. The SDR software generated through EZPro is radio-agnostic and can run on any GPP. This way, we can seamlessly integrate cognitive radio capabilities into tactical radios without any change in radio architecture. Note that once the radio code is generated by EZpro and deployed at a radio, it runs without a need for EZPro. The same code can be used with commercial SDRs for fast prototyping and evaluation of cognitive radio network technologies before implementation on tactical SDRs.

Popular examples of commercial SDRs that can be programmed with EZPro are USRP (Universal Software Radio Peripheral), WARP (Wireless Open-Access Research Platform), bladeRF, HackRF and Matchstiq. Examples of Tactical SDRs are AN/PRC-154 (Rifleman Radio) AN/PRC-152 and WNaN.
that can be programmed using peripheral controller or through implementation on radio processor.

C. SDR Modules for EZPro Implementation

Cognitive radio is driven by a cognitive engine that integrates various network protocols, ideally both adaptive and robust. EZPro has been used for spectrum sensing, DSA, routing, and security functionalities for cognitive radios. Below, we give a summary of cognitive radio network functionalities that can be implemented as modules in the presented SDR programming environment.

1. Spectrum sensing: Cognitive radios discover spectrum opportunities in terms of idle channels via spectrum sensing [8]. A channel can be detected as idle or busy by comparing the accumulated energy of received signal to a predetermined threshold (energy detector) or by exploiting the embedded features in cognitive radio signals (cyclostationary sensing). To prevent poor sensing sensitivity and high sensing error due to device-level energy constraints, poor signal quality (due to fading or mobility) and hidden terminal problem, cooperative (or collaborative) sensing can combining measurements of multiple sensors into one common decision, either by soft combining of channel measurements or by hard combining of binary channel sensing results.

2. Dynamic spectrum access (DSA): PUs (e.g., TV broadcast or radar) have dedicated channels and tactical radios in the role of secondary users (SUs) need to access channels sensed to be available. Else, if there are SUs only working on an unlicensed frequency, it is important not to interrupt existing transmissions. This hierarchical spectrum sharing is typically facilitated by interference avoidance (no interference to Pus and existing SU transmissions is allowed. Channel estimation and neighborhood discovery are two integral parts of DSA to support real-time spectrum monitoring. DSA problem can be also translated to stealth (covert) communications, where tactical radios access the spectrum without being detected by other radios.

3. Cognitive network routing: In a multi-hop network communication setting, spectrum occupancy is time and location-dependent. Therefore joint design of routing and DSA is needed without maintaining any end-to-end path [9]. For example, the backpressure algorithm optimizes a spectrum utility that consists of channel quality (e.g., RSSI) and congestion (e.g., differential queue backlog), and combines routing and channel access decisions in a cross-layer optimization framework.

4. Attack countermeasures: Defense mechanisms to a variety of attacks including jamming SSDF, PU emulation and protocol-violation attacks are needed.
   - Jamming attack: In A2AD environments, tactical radios may receive interference from in or out of network transmission as well as deliberate jammers.
   - SSDF attack: Cooperative sensing makes the cognitive radios vulnerable to SSDF attacks [11], where attackers may flip their sensing results before they report them to the fusion center with the objective of either blocking transmission initiatives or causing transmission failures.
   - PU emulation attack: The goal of this attack is to create an intentional false positive for spectrum sensing by spoofing the role of a PU and generating a waveform similar to that of the PU.
   - Protocol-violation attack: Insider threat arises when users start manipulating the underlying protocols and falsifying the information exchange. These attacks are typically low-signal and aim to degrade the performance slowly over time.

As a defense, cognitive radios can sense spectrum and hop to available channels by tracking and adapting to malicious radio behavior [10] (e.g., by assigning and maintaining trust (or reputation) for each user, monitoring their activities and updating their trusts via consistency check with some type of feedback (e.g., RF signature or protocol behavior)).

III. A Systematic Unified Solution to Cognitive Radio Network Test and Evaluation

Cognitive radio technologies applied to tactical communications should be tested and evaluated extensively before full-scale integration with military systems. Test and evaluation ranges from simulations to the use of over-the-air wireless testbeds and hardware-in-the-loop network emulation tests.

D. Simulations

There are various network simulators, such as ns-2/3, OPNET and QualNet, that can be used to test and evaluate cognitive radio network protocols. They simulate traffic, protocol and PHY effects. More reliable simulator such as CORE and EMANE can be built by using real protocol stack and generating real packet traffic that is carried among nodes represented virtual machines.

E. Over-the-air Wireless Testbed

Simulations use simplistic physical layer modeling that ignores various hardware effects, e.g., nonlinearity, filtering and inter modulation by hardware. There have been several developments of cognitive radio or SDR testbeds e.g., ORBIT in WINLAB [12]. These testbeds can represent static scenarios that cannot be reliably controlled or repeated and cannot emulate a representative geometry of a specific (e.g., mobile) scenario, as often needed to test the refined adaptation nature of cognitive radio functionalities.

F. Emulation Tests

Network channel emulators, e.g., RFnest [3], fill this gap by providing repeatable and controllable testbed for cognitive radios communicating with each other in a dynamic RF
propagation and mobility environment that can be controlled and replayed.

G. Combining Simulation and Emulation Tests

We present a systematic unifying approach, illustrated in Figure 4 to combine simulations and emulation tests, where simulation and emulation tests are controlled under a common scenario by using the same SDR code. In this setup, step 1 is to generate the SDR code (as discussed in Section II); step 2 is to generate and control the scenario; step 3 is to set up the interactions among cognitive network nodes (virtual CORE nodes in simulations and actual radios in emulations) and step 4 is to set up the wireless environment (EMANE for simulations and RFnest for emulations).

![Figure 4](image_url)

(a)

(b)

Figure 4. A systematic unifying approach to combine (a) simulation and (b) emulation tests.

The following steps are followed to simulate a cognitive network solution (Figure 4(a)):

a. EZPro Network Configuration Interface imports the mission scenario from Scenario Generator (available in RFview software) that generates the topology, mobility, channel, platform and radio properties.

b. Using this network scenario, Network Configuration Interface connects to CORE and automatically generates the CORE scenario with virtual nodes where each node represents a specific platform/radio. Network Configuration Interface provides an automated tool to configure each CORE virtual node with standalone software provided by EZPro.

c. Each node in the network is initialized by EZPro with traffic generator and network protocol stack wrapped as a standalone software executable. This software is identical to the one that would be ported to an actual radio and is run on CORE simulator.

d. After this configuration stage, Scenario Generator initiates the CORE simulator that runs virtual nodes with EZPro-generated standalone software.

e. In CORE simulation, EMANE provides PHY/MAC for different waveforms and higher layer network protocols configure tunable knobs in waveforms available in EMANE. Default EMANE parameters can be configured manually by the user before running the simulation or automatically (according to higher layers) during the simulation.

f. Scenario Generator sends channel conditions, mobility and other information (through multicast packet updates) to EMANE. Accordingly, EMANE reconfigures its parameters in near real-time (e.g., < 1ms delay) to simulate wireless network environment.

g. While playing the mission scenario, each virtual node obtains feedback from CORE/EMANE environment. This information is brought to higher network layers.

h. Performance results measured for each virtual node are sent to Scenario Generator with customized GUI to monitor performance (either at network or node level).

To run a cognitive network solution on radios in emulation tests, steps a. and b. are the same as steps a. and b. in simulations, step g. is the same as step h. in simulations and the following additional steps are followed (Figure 4(b)):

c. After this configuration stage, Network Configuration Interface ports network protocols to radio.

d. Scenario Generator sends channel conditions, mobility and other information (through multicast packet updates) to RFnest to emulate wireless network environment.

e. Radio signals go through RFnest between radios and RF channel conditions are adjusted.

f. While playing the mission scenario, each radio learns RF environment (such as spectrum sensing) and this information is brought to higher network layers.

Network control GUI (Figure 5(a)) implemented within EZpro is used to set up the testbed that consists of RFnest, radios connected to the RFnest, and platforms attached to radios (such as the control elements that host the radio software to control the radio). This GUI allows the user to configure nodes (platforms) and radios as virtual or emulated (Figure 5(b)), assign SDR software with virtual nodes in Common Open Research Emulator (CORE) or port and run it to radios.
Network control GUI is hosted on the control station. CORE is used as the network emulator and Extendable Mobile Ad-hoc Network Emulator (EMANE) provides the MAC and physical layers in the virtual tests. The same radio software generated by the presented SDR programming approach (or through other means such as GNU Radio) is used for both virtual and emulation tests.

Transmit power, frequency, signal bandwidth, antenna pattern and gain are some of the configuration parameters for the radios. EMANE model works on the packet level. Signal to Interference and Noise Ratio (SINR) vs. Probability of Reception curves are used to determine whether a packet is successfully received. IEEE 802.11a/b/g and TDMA are example MAC protocols supported by EMANE software.

RFnest is used to emulate wireless channels. The antennas of the radios (e.g., AN/PRC-154A) are removed and the radios are plugged into the RFnest with RF cables. RFnest, digitally controls channel properties between nodes. In this setup, radios do not distinguish whether signals are coming over the air or through RF cables. RFnest can replay channel (I&Q) data and support virtual nodes and interactions with real radios.

RFview software is used to define the test scenarios (Figure 5(c)) including topology, mobility and channel effects. The number and position of the radios are specified on the map. Mobility pattern can be either provided in advance or updated on-the-fly for each radio. The physical layer parameters such as the transmit power or frequency are specified for each radio before starting the tests. RFnest includes various path loss models. The path loss model to be used during the tests is also selected using RFview GUI. The user can create charts on the RFview GUI to monitor the networks performance during runtime. The packet exchanges between the radios can also be visualized on the map during the tests.

H. Field Tests

After the initial T&E of cognitive radio technologies demonstrates the feasibility, the next step is to conduct field tests with tactical radios in realistic and relevant environments before full-scale development and integration. Technology readiness levels increases with each step of T&E and the goal at each step is to demonstrate the improvement of spectrum efficiency increase and mission success. The unified T&E environment for simulations and emulation tests aims to reduce the time from protocol development to field test and increase the success of a field test by evaluating various potential scenarios and identifying problems before the field test.

IV. CONCLUSION

Cognitive radios provide a new communication paradigm to increase spectrum efficiency. This article addressed spectrum challenges for tactical communications and presented a systematic and unified SDR programming, test and evaluation solution with the necessary components to enable cognitive radio solutions for tactical communications. This solution provides fast prototyping of context-aware systematic design along with realistic and relevant experimentation as essential means to achieve the performance gains offered by cognitive radios and ensure network-centric mission success.
Biographies:

YALIN SAGDUYU is Associate Director of Network and Security at Intelligent Automation, Inc. He received his Ph.D. degree in Electrical and Computer Engineering from University of Maryland, College Park. His research interests include wireless networks, cognitive radio and machine learning. He chaired workshops at ACM MobiCom, IEEE CNS and IEEE ICNP, served as Track Chair at IEEE PIMRC 2014 and served in the organizing committee of IEEE GLOBECOM 2016. He is Senior Member of IEEE.

SOHRAAB SOLTANI is Lead Scientist at Intelligent Automation, Inc. He received his Ph.D. degree in Computer Science from Michigan State University. Dr. Soltani’s research interests are in wireless communications, tactical networks, radio programming, cognitive radio, ad-hoc and sensor networks, stochastic networking and wireless system implementation. In this career, he implemented a broad set of wireless (PHY, MAC, network layer) technologies on various radio platforms and executed in-lab and field tests.

TUGBA ERPEK is a Senior Research Engineer at Intelligent Automation, Inc. She received her M.Sc. degree in Electrical and Computer Engineering from George Mason University in 2007. Her research interests include wireless communications, cognitive radio, Dynamic Spectrum Access, network protocols and routing algorithms. She is currently pursuing her Ph.D. degree in Electrical Engineering at Virginia Polytechnic Institute and State University focusing on machine learning for communication systems.

YI SHI is a Senior Research Scientist at Intelligent Automation, Inc. His research interests include algorithm design, optimization, machine learning, communication networks, and social networks. He has been an Editor for IEEE Communications Surveys and Tutorials, a TPC chair for IEEE and ACM workshops, and a TPC member of many major IEEE and ACM conferences. He was a recipient of IEEE INFOCOM 2008 Best Paper Award and IEEE INFOCOM 2011 Best Paper Award Runner-Up.

JASON LI is a Vice President and Senior Director at Intelligent Automation, Inc., where he has been working on research and development programs in the area of networks and cyber security. He received his Ph.D. degree in Electrical and Computer Engineering from University of Maryland, College Park. He led numerous R&D programs related to protocol development for realistic and repeatable wireless networks test and evaluation, moving target defense, and cyber situational awareness.