

# A Platform for Dynamic Spectrum Access Network Experimentation

L.E. Doyle\*, K. Nolan\*, T.K. Forde\*, P. Argryoudis\*, P. Sutton\*, D. Sarath\*, G. Baldwin<sup>†</sup> and M.Ammann<sup>‡</sup>

Centre for Telecommunications Value-chain Research

\*Trinity College Dublin, Ireland

<sup>†</sup>National University of Ireland, Maynooth, Ireland

<sup>‡</sup>Dublin Institute of Technology, Kevin Street, Ireland

**Abstract**—The purpose of this paper is to describe a novel and sophisticated platform for dynamic spectrum access experimentation. The platform comprises, software, hardware and dedicated spectrum. The platform has been designed with experimentation in mind and hence there is a high value placed on the reconfigurable capabilities of the platform. These capabilities are core to providing a wide range of experimentation possibilities and hence the platform does not suggest one particular solution for supporting dynamic spectrum access but facilitates the investigation of multiple solutions. In its broadest sense the platform can therefore be seen as a highly flexible framework that can be populated with information, components, algorithms, methods, logic and intelligence as desired. Currently the platform uses dedicated spectrum that has been set aside by the Irish regulator for software radio experimentation.

## I. INTRODUCTION

The field of dynamic spectrum access focuses on new and more flexible methods for managing spectrum that move away from the traditional command and control means of regulation. Secondary markets in spectrum, opportunistic use of white spaces, easement usage and open spectrum solutions are among the many different approaches that can be taken to create a world in which spectrum is more effectively and efficiently used. The path away from the command and control world requires advancements to be made both on the technology and policy fronts as well as mindset changes in the relevant communities. Experimental platforms that explore the many technical issues associated with the new regimes as well as demonstrate the feasibility of various solutions are an essential part of this process.

There are a number of platforms currently in existence that can be broadly classed as platforms for dynamic spectrum experimentation. They typically have come about under the heading of cognitive radio research with dynamic spectrum access/management highlighted as being a key application for cognitive radio. It should be noted that there are some other experimental platforms that can be classed as general software radio platforms but are not included here as the emphasis in this paper is on those platforms that specifically gear themselves towards dynamic spectrum experimentation.

The Cognitive Wireless Technology group (CWT) in Virginia Tech is actively involved in the development of a cognitive engine system for multi-band and multi-mode wireless applications [1], [2]. Their approach uses case-based decision

theory and genetic algorithms to update and optimize radio parameters. The cognitive engine includes three modules, comprising the modeling blocks (REM-Radio Environment Modeler, performance and hardware monitor, user and policy modeler), the Wireless System Genetic Algorithm (WSGA) module for optimizing radio adaptation and the Cognitive System Monitor (CSM) module for intelligent reasoning and decision-making. The group are planning a network of cognitive radios with over a thousand nodes [3].

The Mobile and Portable Radio Research Group (MPRG) at Virginia Tech, developed the OSSIE (Open Source SCA Implementation: Embedded) testbed to facilitate research in the field of software defined radio (SDR) [4], [5]. Their cognitive radio testbed consists of a combination of Tektronix off-the-shelf components and OSSIE and it is used to test and validate cognitive radio ideas. The MPRG take a game theory approach to cognitive radio and the testbed facilitates the analysis of radio etiquettes developed from their game theory research and the stability and convergence of the cognitive algorithms they design.

The Cognitive Radio Research (CORR) team at Berkeley Institute has developed the BEE2 (Berkeley Emulation Engine 2) [6]. The BEE2 was built as a generic, multi-purpose, FPGA-based emulation platform for computationally intensive applications. It has been targeted at cognitive radio applications and in particular to compare sensing techniques for opportunistic radio use. A testbed based on the multi-FPGA emulation engine BEE2 can connect to 18 radio front-ends, which can be configured as Primary or Secondary Users and the various approaches to sensing and the resulting interference evaluated [7].

DARPA XG (Defense Advanced Research Programs Agency Next Generation Program) focuses on developing technology to allow multiple users to share use of the spectrum through adaptive mechanisms that deconflict users in terms of time, frequency, code, and other signal characteristics [8]. The project aims to gain ten times more spectrum efficiency without interference to non-XG radios and intends to demonstrate these capabilities in military and urban RF environments. One of the focuses within the project is on the development of policy languages for spectrum management. A testbed of 25 nodes is planned for the project.

While these and other testbeds greatly add to the body

of knowledge in the field, there is a clear need to further expand the practical and experimental work in the area of dynamic spectrum access and management. The experimental platform described in this paper brings with it a unique set of features that can explore and push the field further. The platform was designed following wide experience in software radio systems [9], [10]. The platform has been designed to be as flexible as possible, to facilitate a very high degree of software reconfigurability, to embrace multiple approaches to dynamic spectrum access and in the longer term to be available over a web interface to those who wish to use it. Currently the platform has its own dedicated spectrum for software radio and dynamic spectrum experimentation.

The paper is organized as follows. Section II describes the requirements of the platform and introduces the platform briefly. The two main parts of the platform are the Reconfigurable Node and the Cognitive Wrapper. The Reconfigurable Node is described in section III followed by section IV which briefly looks at the hardware frontend of the node and the associated experimental license. The Cognitive Wrapper is discussed in section V. Section VI focuses on the next stage in the development of the testbed and section VII concludes.

## II. THE ESSENCE OF DYNAMIC SPECTRUM ACCESS SYSTEMS

To design a platform that can be used for a wide range of experimentation, it is necessary to look at the essence of what is needed in a network that supports dynamic spectrum access. In very general terms a node in such a network must have three broad capabilities. Firstly it must have the ability to *observe* all it can know about its environment and the context within which it is operating. Secondly it must have the ability to *decide* how to respond to any changes it has sensed. And finally it must have the ability to *act* in response to the decisions. The *observe, decide, act* cycle can be seen as a more simplified representation of Mitola's cognition cycle [11].

### A. Observe

In terms of dynamic spectrum applications the observations that are most typically needed are those relating to the availability of spectrum and the identification of white spaces. However it is useful to think beyond the spectrum sensing problem and think of all the changes a node in a network might need to sense or observe to facilitate dynamic spectrum access. We have defined seven classes of changes that are relevant and they are:

- 1) Radio Environment Changes (e.g. changes in communication channels conditions, increased/decreased noise, changes in availability of spectrum etc.)
- 2) Network Environment Changes (e.g. changes in the density, degree, mobility, size, membership of the network etc.)
- 3) Regulatory Environment Changes (e.g. movement from US to EU, change from liberal to non-liberal jurisdiction, etc. )

- 4) Application Requirement Changes (e.g. changes in bandwidth requirements, changes in data rates needed, changes in processing power needed etc.,)
- 5) Physical Environment Changes (e.g. changes from day to night, rural to urban, etc.)
- 6) Business Environment Changes (e.g. changes in tariffs, changes in willingness to pay, changes in level of competition available, upgrades etc.)
- 7) Social Environment Changes (e.g. changes in social demands (public safety needs), changes from individual to cooperative behavior patterns)

All of the changes enumerated here can trigger the seeking out and use of alternative spectrum, whether it be to improve communication conditions because of bad signal-to-noise ratios in given bands or because of a change in regulatory jurisdiction or because of the need to get a better deal etc. *Therefore a platform that supports dynamic spectrum access experimentation needs to be able to observe and sense in the widest meaning of the terms.*

### B. Decide

The next step is to make a decision as how to react to observed changes. Decisions can fall into two main categories; unilateral decisions that are taken by a node alone and multilateral decisions that must be taken together by all involved. Decisions of course must typically be made within a given set of constraints. So for example there is no point in deciding to reconfigure to use a different band of frequencies, if the frequency agile frontend hardware is not capable of supporting the chosen band. Decisions can be made in a variety of ways that range from using simple reasoning to making high-level cognitive deductions. For example a fully cognitive network would have the ability to figure out what is necessary, possible, desirable, permitted, most suited etc. in response to a set of changes. And it would have the ability to learn and change its reactions with experience. *Whatever the approach to decision-making, a platform that supports dynamic spectrum access experimentation needs to be able to facilitate unilateral and multilateral decision-making within the constraints of the context in which it is operating.*

### C. Act

The ability to act manifests itself as the ability of a node in a network or all nodes in the network to reconfigure. Reconfiguration should not be considered in terms of the physical layer (i.e. software radio) only. To facilitate all possible actions and reactions that are needed, nodes must support *complete* hardware and software reconfiguration. By complete reconfiguration we mean reconfiguration from application layer to physical layer at multiple speeds and on multiple levels (i.e. reconfiguration of one parameter of a layer would be a low level reconfiguration whereas reconfiguration of the entire node would be a high level reconfiguration). The speed and granularity of reconfiguration in reaction to the type of changes articulated above can vary widely. For example a reconfiguration of the modulation scheme to match

an observed degradation in signal-to-noise level can be instantaneous and can occur repeatedly as the signal level increasing deteriorates. On the other hand a reconfiguration to comply with regulatory policies in a particular geographical area may take longer and involve a whole system change but only occur once on entering that jurisdiction. *Therefore a platform that supports dynamic spectrum access experimentation needs to be flexible enough to support highly varied speeds and granularities of reconfiguration.*

#### D. Platform

Our platform provides a means of fulfilling the demands of the observe, decide, act cycle. The platform is a collection of hardware and software components. The software components are entirely general-purpose processor based and make up the vast bulk of the system with a limited amount of hardware used. The hardware components, i.e. the RF frontend, are connected to the PC with a suitable interface. The complete platform can be divided into two parts. The first part is called the **Reconfigurable Node** and around the Reconfigurable Node is the second part, the **Cognitive Wrapper**. These are depicted in Figure 1.

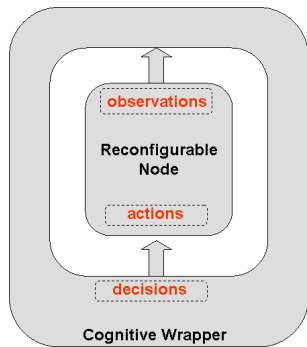


Fig. 1. A Conceptual Representation of the Experimental Platform

The Cognitive Wrapper directs the node how to act (i.e. it decides how to respond to changes) and the node itself is responsible for implementing the decisions as well as making/percuring the necessary observations (i.e. it observes and acts). In section III the Reconfigurable Node is described. The following section, section IV focuses on the current hardware associated with the Reconfigurable Node before returning in section V to the Cognitive Wrapper.

### III. THE RECONFIGURABLE NODE

In very broad terms we view a node in the network as a structure comprised of heterogeneous software and (a few) hardware components. These components are linked together to form a desired node configuration and they may be dynamically altered as and when is needed. A detailed description of the Reconfigurable Node can be found in [12] but the main aspects of the system are included in this paper.

Components in the Reconfigurable Node tend to have greatly varying levels of granularity. For example, a component may encapsulate an entire node layer (such as the

Network or Physical layer of the OSI model) within a network stack architecture or may contain a subsection of such a layer (such as a filter or digital modulator). Each component will have a number of associated parameters which control its operation (such as the cache timeout of an ad-hoc routing layer or the tap values of a filter). A significant advantage of adopting such a component-based design is the ability to manage a very high degree of complexity. By encapsulating the implementation of specific parts of the node in components, it is possible for a node designer to effectively build a wide range of complex systems without detailed knowledge of every aspect of each component used. To facilitate this we have designed a universal component container or shell which is used to wrap the unique functionality of each heterogeneous component and allow it to be handled in a generic manner.

#### A. Components

Two specific component types make up the fundamental building blocks of the Reconfigurable Node. These are the Layer Component and the Radio Component. The Layer Component is, as it says, a layer of the stack. The Radio Component is typically the type of component which features within the physical layer and is used to encapsulate a single signal processing step of an isochronous signal chain. A number of interfaces have been designed to control the components. The *Inter-Component Interface* permits components to be linked together to form a network stack structure and facilitates the flow of data upwards and downwards over this link. The *Component Lifecycle Interface* is provided to allow an external controller to start, suspend, resume and stop the operation of the component as well as initialize and destroy the component itself. The *Parametric Reconfiguration Interface* facilitates the manipulation of a component's operating parameters in a generic manner and the *Component Event Interface* is used to notify an external listener of any significant occurrences within the component. The relevance of each of these interfaces will become obvious in later sections.

#### B. Framework

Figure 2 illustrates the principal elements of the framework involved in the process of creating and managing the node network stack. These elements are the Stack Manager, the Configuration Parser and the Component Manager, each of which is presented in turn below.

The key controlling entity of the reconfigurable node is the Stack Manager. A user of the system creates a radio/node configuration of choice by stringing a set of components together. If the desired component does not exist in our inventory of components, then a new component must be created by the user. A particular radio/node implementation is defined in a configuration file which is a simple XML file. Given a radio/node configuration, the Stack Manager identifies, creates and initializes the required components. The components are then linked together to form the structure of the node's network stack and the operation of each is started. Throughout the operating period of the network stack, the Stack Manager is

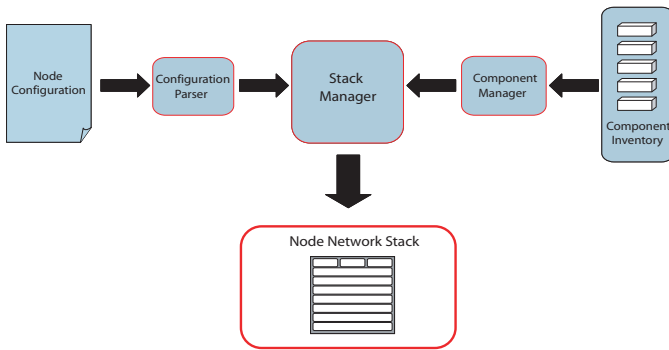


Fig. 2. The Management Elements

responsible for handling and reacting to any events signalled by the components as well as performing any reconfiguration required. When the operation of the network stack ceases, the Stack Manager halts the operation of each component, unlinks them to break up the stack structure and destroys them.

In order to carry out these tasks successfully, the Stack Manager uses a Configuration Parser to handle the parsing of the XML configuration file. The Configuration Parser is the element of the reconfigurable radio with the responsibility of translating the network stack configuration into a lightweight data structure which can be read by the Stack Manager. The Configuration Parser also provides the ability to create a full XML configuration document from the stack data structure and thus capture the stack state at any time. The system uses what is known as the Component Manager to manage the components. The Component Manager maintains an inventory of components which may be used by the Stack Manager to build the node network stack.

### C. Making Observations

To ability to make observations such as those discussed in section II is very important. Two observation mechanisms for fulfilling this role are provided. Together these mechanisms permit observations to be both pushed to the Cognitive Wrapper by the Reconfigurable Node and pulled from the Reconfigurable Node by the Cognitive Wrapper. The first of these mechanisms allows any component within the node network stack to notify the Stack Manager of a significant occurrence. These Events may be defined by the designer of the individual component and can be used to highlight any occurrences that the designer considers of use to the Cognitive Engine. Examples include the crossing of a given threshold for the bit error rate within a Reed-Solomon decoder component or for the link change rate of an ad-hoc routing component. By signalling a given event, the component makes the Stack Manager aware of the specific occurrence and allows it in turn to inform the Cognitive Wrapper.

The second mechanism, referred to above, permits observations of the outside world to be pulled from the Reconfigurable Node by the Cognitive Wrapper. This mechanism consists of a shared database referred to as the Reconfigurable Node Blackboard. The Blackboard permits each of the components

of the network stack to expose any information which may be used to inform decisions made by the Cognitive Engine. Examples of such information could be the power spectral density obtained at the FFT stage of an OFDM receiver or the routing table of a network layer routing protocol. By providing both a push and pull mechanism for the exchange of observations between the Reconfigurable Node and the Cognitive Engine, both reactive and proactive approaches to decision-making may be facilitated.

Two type of observations are currently used within the system. The first is what we refer to as a *dedicated* observation. The second type is and *opportunistic* observations. A dedicated observation occurs for example when the system is used to implement a cyclostationary feature detector. In this case the node itself has an actual dedicated sensor within its structures and such a dedicated sensor would typically notify the Stack Manager of a significant occurrence and it would expect a certain response. The opportunistic observations are those which happen during the normal course of events such as a routing layer noting the numbers of neighbors it has. These opportunistic observations are typically placed on the Blackboard to be used as the Cognitive Wrapper so chooses.

### D. Implementing Actions

In general three types of actions (i.e. reconfigurations) are possible within the platform. The first, **parametric reconfiguration**, involves the dynamic alteration of individual parameters of any of the components (e.g. change of a filter coefficient, change of cache size in routing tables). The second, **structural reconfiguration**, involves the alteration of some component of the node while still performing the same overall application (e.g. change of modulation scheme, change of MAC layer). The third, **application reconfiguration** involves completely replacing the software of the node with an entirely different Reconfigurable Node configuration (e.g. change from WiMAX to WLAN).

At the most fundamental level of reconfiguration is the ability to alter individual parameters within each layer of a structure and to this end the Parametric Reconfiguration Interface (PRI), as described in section III-A is used. By specifying the reconfigurable parameter label and providing an associated value, the PRI may be used in a generic manner to alter parameters of any layer in a given configuration. Although this interface is common to each layer, the reconfigurable parameters are layer-specific so a process involving metadata and automatic code generation is used to provide the specific functionality needed at each layer. A dedicated script uses metadata tags inserted in the layer code to generate the reconfiguration code required. This code is then included in the layer implementation.

Two additional common component interfaces play vital roles in facilitating structural reconfiguration of the node network stack. Firstly the Inter-Component Interface (ICI), as described in section III-A has a role. This forms the link between adjacent components of the network stack. In order to permit alteration of the network stack structure, it is essential

that this ICI can be rapidly torn down and re-established. With these requirements in mind, a layer-neutral packet structure was implemented to encapsulate the data flowing between components of the network stack. The use of this packet structure greatly reduces the complexity needed for the ICI and simplifies the process of linking and unlinking components to form the node network stack structure. Secondly the Component Lifecycle Interface (CLI), also introduced in section III-A, is used to externally control the operation of the individual component and is essential in the process of structural reconfiguration. As the links between components are altered, the CLI is used to suspend and resume component operation in order to avoid potential conflicts, as well as to start the operation of inserted components and stop that of components which are being removed.

The Stack Manager, introduced in section III-B is a key enabling feature of full application reconfiguration of the reconfigurable core of the platform. The description in section III-B of how the stack is created and started by the Stack Manager is also how the Stack Manager reconfigures the stack. In order to permit external reconfiguration of the stack, the Stack Manager provides a *Stack Control Interface* (SCI) which acts as an API for the reconfigurable core. Using the SCI, an external controller (e.g. the Cognitive Wrapper) may trigger a reconfiguration by passing a new XML configuration to the Stack Manager. This configuration is parsed and the resulting data structure compared with that of the existing stack. Each difference between the two will trigger a reconfiguration on one of the levels discussed above, the sum total of which may result in a completely new stack structure and application.

#### IV. THE HARDWARE FOR THE RECONFIGURABLE NODE

The Reconfigurable Node is designed to be used with any hardware of choice. To interface to the hardware, a writer and reader component must be created. These software components are the last software components in the Reconfigurable Node before the physical hardware. Currently the hardware in use has been designed for a particular test license. While details of this are given here it should be remembered that this is one example of many suitable hardware frontends.

##### A. The License

Before discussing the hardware it is useful to briefly discuss the operating frequencies of the system. The Irish spectrum regulators, ComReg<sup>1</sup>, granted the research team an experimental licence for software radio and dynamic spectrum management. The licence allows for experimentation to take place in two 25MHz bands, one centered at 2.08GHz and the other at 2.35GHz. The transmit power is 1 Watt EIRP and the license can be used around three major cities in Ireland (Dublin, Cork and Limerick). The relative availability of spectrum in Ireland and the fact that Ireland is an island, and as such cross-border interference is limited, has facilitated this in part. The openness of the regulators has also aided the research. The interface for

remote control of the platform therefore allows international, or geographically remote, researchers to carry out experiments with Irish spectrum.

##### B. RF Frontend

The philosophy behind the hardware developed here has been to keep as much of the processing capability on the general-purpose processor as possible. Figure 3 shows the schematic of the circuit involved. To this end the hardware consists of a USB interface with an embedded 8051 micro-controller, two 16 bit ADCs and two 16 bit DACs at the baseband. These form a quadrature baseband system. These are connected to the receiver and transmitter radios respectively by a pair of reconfigurable lowpass filters. On the receive side the radio is implemented as a direct conversion I/Q architecture using off-the-shelf components. On the transmit side the radio is again implemented as a direct conversion I/Q architecture. The direct conversion layout was chosen to allow the maximum flexibility in choice of transmit and receive frequencies with out recourse to banks of filters that would be required in an equivalent heterodyne system.

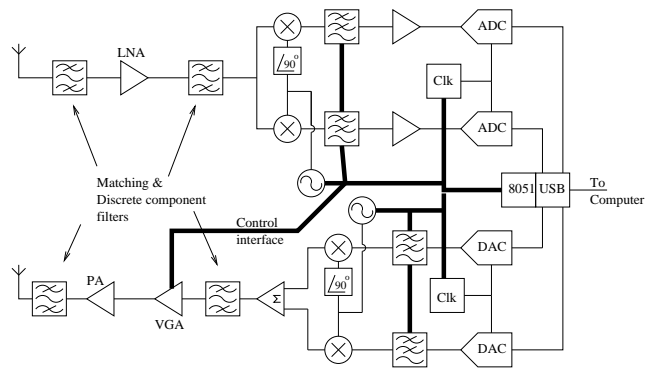


Fig. 3. Hardware Frontend Schematic

The ADCs, DACs and USB interface FIFOs are clocked from a reconfigurable clock source on the board. It is programmed over an I2C interface from the embedded micro-controller. Similarly the local oscillator in both the transmitter and the receiver are also fully programmable. These are again controlled over an SPI interface from the micro-controller. Finally the transmitter power is controlled using a variable gain amplifier in the transmit path, which is again controlled using the data bus of the embedded micro-controller.

The hardware is designed to meet among others the GSM specifications for pico-basestations and can operate in the frequency range 1.6GHz to 2.5GHz. Within this range it can be used for a range of protocols and frequency bands including 2G, 3G WiFi and WiMAX. The current capability of the system is only restricted by the available data bandwidth over the USB bus. The transmit path has a maximum data conversion rate of 100Msamples per second in both the I and the Q channel. This is backed by a 160MHz maximum bandwidth in the rest of the transmitter chain. The receive path is also capable of 90MHz in both the I and Q channels. This

<sup>1</sup>ComReg - Commission for Communications Regulation (<http://www.comreg.ie>)

feeds into a pair of ADCs capable of sampling at a maximum rate of 130Msample per second. The receiver has a designed -109dBm reference sensitivity while the transmitter is capable of transmitting up to 1 watt with eight bits linear resolution of power control. This configuration is capable of meeting most of the mobile communication standards in use to day and provides a flexible platform for the development of future standards and also the exploration of software defined radio.

The reconfigurable node has software components that write to and read from the hardware and can send commands to the frontend to control operation.

### C. Antenna Design

To make use of the licensed spectrum an antenna was designed for the bands of interest, i.e.to center on frequencies of 2.08 and 2.35 GHz. The printed strip monopole antenna is printed on one side of a FR4 substrate with the groundplane located at the back as depicted in Figure 4.

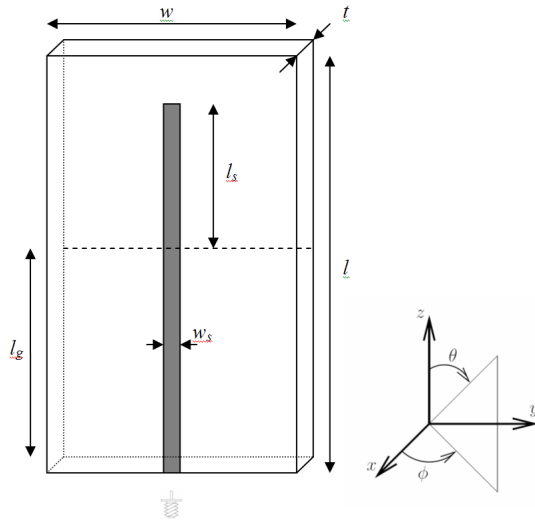


Fig. 4. Antenna Geometry

The dimensions of the substrate are  $l=70\text{mm}$  and  $w=35\text{mm}$ . The groundplane is  $35\text{ mm}^2$ . The radiating element is a microstrip line ( $l_s = 30\text{mm}$ ,  $w_s = 2\text{ mm}$ ) which is excited by an SMA connector. The substrate is  $1.52\text{ mm}$  thick and the metallization thickness is  $35\text{ microns}$ .

The radiation patterns for the antennas are as shown in Figure 5 and the measured return loss is given in Figure 6.

The antenna gain is  $2.2\text{ dBi}$  across the band  $2.0\text{ GHz}$  to  $2.35\text{ GHz}$ .

### V. COGNITIVE WRAPPER

The Cognitive Wrapper is the part of the system that directs the Reconfigurable Node. It is associated with the decide part of the *observe, decide and act* cycle discussed in section II. In other words the Cognitive Wrapper controls the software and hardware components of the Reconfigurable Node. The main elements of the Wrapper are shown in Figure 7 and consist of the (A) decision-making engine, (B) the constraint representation structure and (C) the knowledge matrix.

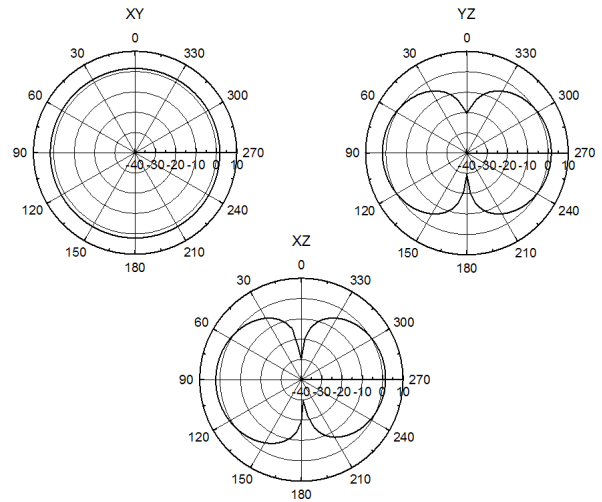


Fig. 5. Radiation patterns at 2.215GHz for xy, yz and xz planes

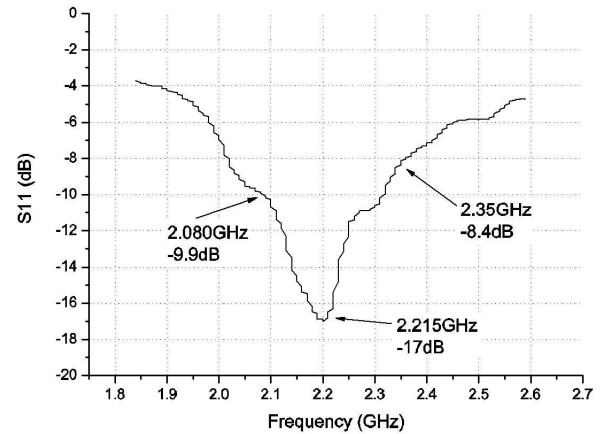


Fig. 6. Measured return loss

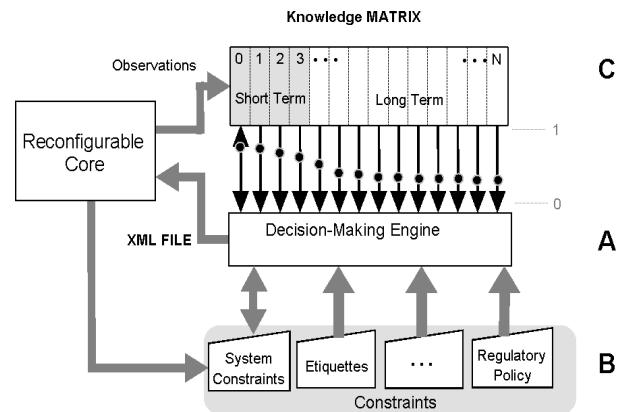


Fig. 7. Cognitive wrapper overview illustrating the knowledge representation delay-line, reasoning and learning engine, external inputs and reconfigurable entity

### A. Decision-Making Engine

At the center of the wrapper is the decision-making engine. Any feasible simple logic, game-theoretic, genetic algorithmic, artificial neural networks, Bayesian or Fuzzy system logic implementation approach can be used here [13], [14], [15], [16]. The feasibility of a decision-making approach is dependent on the time required to present viable solutions and the implementation complexity associated with each approach. The presentation of a solution approaching optimality within the time constraints allowed has a potentially greater value than an optimal solution that is produced too late i.e. after the implementation deadline. Complexity and the processing burden can be reduced by implementing some features of a chosen approach. It is conceivable that significant gains using dynamic spectrum access techniques can be achieved without the full weight of a maximal-complexity cognitive engine. The feasibility of the decision-making approach also depends on the type of decision that needs to be made. As stated in section II some decisions will be standalone unilateral decisions made by the node whereas others will be network-wide decisions in which either a controlling entity (e.g. basestation) directs procedures or a distributed decision-making process ensues. Figure 8 attempts to capture diagrammatically these points.

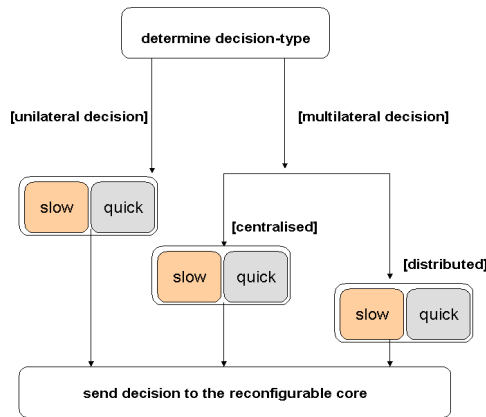


Fig. 8. The Decisions which need to be made about Decision-Making

It should be noted that the terms quick and slow in Figure 8 are themselves relative to the decision that is being made.

Irrespective of the approach used, once the output is an XML file, the file can be passed on to the Reconfigurable Node for implementation. Recall that using the SCI, described in section III-D the decision-making engine can trigger a reconfiguration by passing a new XML configuration to the Stack Manager. In fact any decision-making engine designed elsewhere can interface with the Reconfigurable Node via an XML file.

The whole decision-making engine can also be reduced to a lightweight structure. Recall in section III-C that one of the mechanisms for signalling the occurrence of an event of interest allows any component within the Reconfigurable Node to notify the Stack Manager. Rather than involve complex reasoning and decision-making the event can also trigger a pre-

selected response. So for example should the cyclostationary feature detector mentioned in section III-C identify unambiguous white space, then the response could be reconfigure hardware frontend and use the spectrum.

### B. Constraint Representation

Any decision must be made within a set of constraints. There may be regulatory constraints within given geographical areas. There may be hardware constraints for the system itself (e.g. frequency bands in which it can operate, CPU or battery capabilities). There may be financial constraints in terms of what a user is willing to pay for. The platform makes use of the idea of policies to encode any constraints (i.e. restrictions, behaviors and preferences etc.). This approach is used in a number of the platforms mentioned in section I. To keep with the general nature of the approach we do not assume that all policies are local to any given node in the system and may at times have to be remotely accessed.

To illustrate how this is used we have prepared policies for using our test license. The policy simply states that there are two bands (BandA and BandB) corresponding to the two bands of the license as well as six operations that can apply on the bands. It allows any subject (requester) to perform any of these six operations on either band if she has an email address in one of the seven domains in which the license can be used. The request can be evaluated against the policy using any XACML implementation (for example <http://sunxacml.sf.net/>).

XACML (eXtensible Access Control Markup Language) is an XML-based language for access control. XACML describes both an access control policy language and a request/response language. The policy language is used to express access control policies (who can do what when). The request/response language expresses queries about whether a particular access should be allowed (requests) and describes answers to those queries (responses). Figure 9 gives an indication of the content of one of the policies.

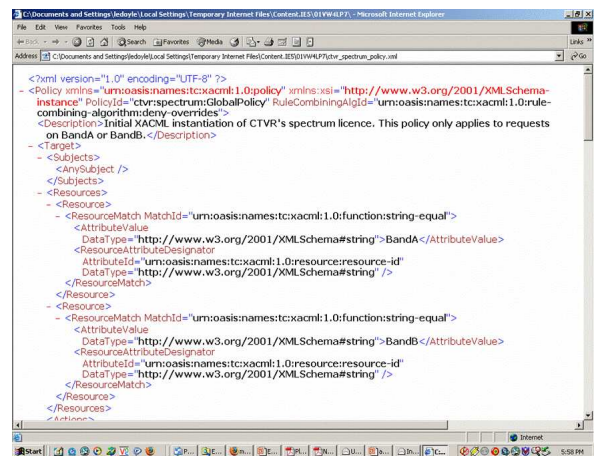


Fig. 9. A Screen Grab of the Policy for the License

Any constraints can be dealt with in this manner. So for example it is easy to imagine that a node on entering a

given jurisdiction would automatically download the local regulatory policy from a policy server or get the equivalent of a SIM card containing the policies to be heeded. Likewise any Reconfigurable Node can produce a set of policies that broadly outline the limits of its operation. Various policies may of course conflict. There is no mechanism in the current version of the testbed for dealing with conflict but this is being addressed and will be dealt with in future versions.

### C. Knowledge Matrix

As stated above the decision-engine makes a decision based on a set of observations and under given constraints. The knowledge matrix in Figure 7 is a means of dealing with the observations that are fed to the decision-making engine. More extensive details of this can be found in [17].

In simple terms the knowledge matrix manages observations. The observations are fed to the matrix from the Reconfigurable Node. Either the node itself has been tasked with making dedicated observations or the observations are obtained via the node from elsewhere. This allows for observations to be bought, collectively and collaborative determined or obtained from some other external source. The knowledge matrix facilitates the ability to store, order, extract and reuse information relating to current and historical state, actions, conclusions, objectives in a structured format.

The knowledge matrix is divided into memory delay lines. The memory delay line is the means used to store current and historical knowledge sets for the decision-making engine. The memory delay line is analogous to a finite-length filter delay-line which stores current and historical knowledge sets. Information from all stored memory sets is available for use by the decision-making engine. The relevance of certain aspects of each knowledge set stored in the memory delay-line may not be constant. A memory-merging capability offers some interesting possibilities. For some scenarios, identification of common traits, actions or consequences of previous radio reconfiguration and observed events may be more important than spurious events or actions. Selective memory can also be used to place a greater bias on recent knowledge rather than longer-term knowledge, or vice-versa. Weighting factors, analogous to filter coefficients are used to implement memory selectiveness. The selective nature is reconfigurable by varying the weighting factors associated with the knowledge set stored in each memory delay. The two extreme cases of (1) a memory-less radio device is achievable by assigning a weighting factors of zero for all memory delayline weights and (2) a photographic memory is achievable by assigning a weighting factor of one for all memory delayline weights and averaging the result.

## VI. FUTURE WORK

The experimental platform version 1 as described in this paper is in use. However only basic tests have been carried out to date as currently only two nodes exist. There is a means of having a mixture of real and simulated nodes in the system. This is equivalent to a reconfigurable node having two

frontends, one real and one a (simple) simulation layer. This currently addresses the need for more nodes, but more real nodes are essential. The next stage of the process will involve an extensive test and experimentation regime featuring more nodes using the test license in a variety of ways.

In an effort to make the testbed more widely available, a web interface over which it can be controlled is currently being designed. In the short term this interface will allow simple set up and control of experiments using the existing features. In the longer term it is hoped that it will be possible for interested parties to insert their own algorithms for test.

One of our major interests is exploring market-based approaches to spectrum assignment. We envisage using the testbed as a partial means of exploring the issues involved. The platform will be extended to include mechanisms for dealing with authorization and payment so that spectrum can be traded and paid for. We use an authorization scheme based on a security management system developed in our research group known as *ÆTHER* [18]. Although *ÆTHER* has been designed to address trust establishment and access control in pervasive computing environments, we believe that the highly distributed and disaggregated systems we design on our testbed can benefit from its flexible management framework. *ÆTHER* directly extends the traditional Role-Based Access Control (RBAC) model [19] to define a framework that supports decentralized administration, disconnected operation and context-awareness. Furthermore, it uses the well-defined concept of location-limited channels [20], [21] to specify an unobtrusive usage model for the required administrative tasks. Based on this general framework it instantiates two different systems. The first one, *ÆTHER*<sub>0</sub>, has been designed to address the authorization requirements of small environments which consist of particularly constraint devices, like 8-bit sensors, and utilizes only symmetric cryptography. The second, *ÆTHER*<sub>1</sub>, addresses the authorization requirements of large computing domains that have multiple owners with complicated security relationships. This greater flexibility comes at the cost of using asymmetric cryptography which is more computationally expensive. More details can be found at [22].

According to our vision the use of available spectrum can be viewed as a service offered by a provider to potential clients. As we move to highly mobile and decentralised usage paradigms, roaming users must be allowed to access the spectrum offered by independent third-party providers in a dynamic manner. The use of this provided spectrum must be accounted for in *real-time* in order to facilitate the mobility patterns and the needs of roaming clients. Existing electronic payment systems can be used to support this requirement. We can broadly classify such systems into two categories; *macropayment* systems and *micropayment* systems. After a careful examination of the properties of both categories we have concluded that the spontaneity, the disconnected operation and the need for infrequent interactions with brokers that micropayment systems demonstrate make them the most appropriate for our testbed.

Our group has developed a micropayment scheme for mo-

bile networks based on hash chain trees [23] that can be utilised for implementing the payment protocol between a client and a spectrum provider. We plan to use the  $\text{\AE THER}$  security management framework to facilitate the micropayment authorisation of spectrum use. With  $\text{\AE THER}$  we can specify:

- a) the conditions under which a spectrum provider accepts payments,
- b) a user's contract with a broker that provides digital coins, and
- c) the terms of spectrum provision between a user and a provider.

Once authorization and payment components are available within the testbed, various different trading strategies can be explored.

## VII. CONCLUSION

The testbed presented in this paper is a highly flexible system that can be used to facilitate a wide range of experiments related to dynamic spectrum access and management. The testbed is highly flexible and allows a very large degree of reconfiguration possibilities. Testbeds are an essential part of the process in designing, developing and testing new regimes for accessing and managing spectrum.

## ACKNOWLEDGMENT

The research was made is supported by Science Foundation Ireland under grant no. 03/CE3/I405 .

## REFERENCES

- [1] D. Maldonado, B. Le, A. Hugine, T. W. Rondeau, C. W. Bostian, "Cognitive radio applications to dynamic spectrum allocation: a discussion and an illustrative example," *IEEE Proc. DySPAN 2005*, 8-11 Nov. 2005, pp. 597 - 600.
- [2] Rondeau T, Bin Le, Maldonado D, "Introduction to the Cognitive Engine", Project Meeting at Virginia Tech, September 2005.
- [3] Charles W Bostian, "An Opportunity to build a large scale Cognitive Wireless Network", Presentation at Virginia Tech, January 2006.
- [4] [http://www.mprg.org/research/cognitive\\_radio/index.html](http://www.mprg.org/research/cognitive_radio/index.html)
- [5] <http://ossie.mprg.org/>
- [6] <http://bee2.eecs.berkeley.edu/>
- [7] Mishra SM, Cabric D, Cheng Chang, Willkomm D, van Schewick B, Wolisz A, Brodersen RW, "A Real-Time Cognitive Radio Testbed for Physical and Link Layer Experiments", *IEEE Proc. DySPAN 2005*, 8-11 Nov 2005, pp562-567
- [8] Website of Next Generation Program, ATO (Advanced Technology Office) <http://www.darpa.mil/ato/programs/XG/index.htm>
- [9] Philip Mackenzie, "Software and reconfigurability for software radio systems," *Ph.D dissertation*, Trinity College Dublin, Ireland, 2004.
- [10] Keith Nolan, "Reconfigurable OFDM Systems," *Ph.D dissertation*, Trinity College Dublin, Ireland, 2005.
- [11] Mitola, J., III; Maguire, G.Q., Jr., "Cognitive radio: making software radios more personal," *Personal Communications, IEEE [see also IEEE Wireless Communications]* , vol.6, no.4pp.13-18, Aug 1999
- [12] Sutton, P., Doyle, L., Nolan, K.E., "A Reconfigurable Platform for Cognitive Networks", in *Proceedings of the 1st International Conference on Cognitive Radio Oriented Wireless Networks and Communications (CROWNCOM 2006)*, June 8-10, 2006, pp6.
- [13] J. Neel, J. Reed, R. Gilles. The Role of Game Theory in the Analysis of Software Radio Networks, SDR Forum Technical Conference November, 2002.
- [14] Rieser, C.J.; Rondeau, T.W.; Bostian, C.W.; Gallagher, T.M., "Cognitive radio testbed: further details and testing of a distributed genetic algorithm based cognitive engine for programmable radios," *Military Communications Conference*, 2004. MILCOM 2004. *IEEE* , vol.3, no.pp. 1437- 1443 Vol. 3, 31 Oct.-3 Nov. 2004
- [15] Hopfield, J.J., "Artificial neural networks," *Circuits and Devices Magazine, IEEE* , vol.4, no.5pp.3-10, Sep 1988
- [16] Klir, G.J., "Fuzzy logic," *Potentials, IEEE* , vol.14, no.4pp.10-15, Oct/Nov 1995
- [17] Nolan, K.E., Sutton, P., Doyle, L., "An Encapsulation for Reasoning, Learning, Knowledge Representation, and Reconfiguration Cognitive Radio Elements", in *Proceedings of the 1st International Conference on Cognitive Radio Oriented Wireless Networks and Communications (CROWNCOM 2006)*, June 8-10, 2006, pp6
- [18] P.G. Argyroudis, and D. O'Mahony, " $\text{\AE THER}$ : an Authorization Management Architecture for Ubiquitous Computing", in *Proceedings of 1st European PKI Workshop: Research and Applications (EuroPKI'04)*, June 2004, vol. 3093 of Lecture Notes in Computer Science, pp. 246-259, Springer-Verlag.
- [19] D.F. Ferraiolo, and D.R. Kuhn, "Role-Based Access Controls", in *Proceedings of NIST-NSA National Computer Security Conference*, 1992, pp. 554-563.
- [20] F. Stajano, and R. Anderson, "The Resurrecting Duckling: Security Issues for Ad hoc Wireless Networks", in *Proceedings of 7th International Workshop on Security Protocols*, 2000, pp. 172-182.
- [21] D. Balfanz, and D.K. Smetters, "Talking to Strangers: Authentication in Ad hoc Wireless Networks", in *Proceedings of 9th Network and Distributed System Security Symposium (NDSS'02)*, 2002.
- [22] P.G. Argyroudis, *Authorization Management for Pervasive Computing*, Ph.D. Thesis, Department of Computer Science, University of Dublin, Trinity College, 2005.
- [23] T. Hitesh, and D. O'Mahony, "Real-Time Payments for Mobile IP", *IEEE Communications*, vol. 41, no. 2, pp. 126-136, 2003.