Advanced coexistence technologies for radio optimisation in licensed and unlicensed spectrum

(ACROPOLIS)

Document Number D7.1

Evaluation and Roadmap of Existing SDR, CR Platforms and Future CR Systems

Contractual date of delivery to the CEC:	30/09/2011
Actual date of delivery to the CEC:	30/09/2011
Project Number and Acronym:	257626 - ACROPOLIS
Editor:	Sylwia Romaszko (RWTH), Marina Petrova (RWTH)
Authors:	Sylwia Romaszko (RWTH), Petri Mähönen (RWTH), Marina Petrova (RWTH) Junaid Ansari (RWTH), Xi Zhang (RWTH), Elena Meshhova (RWTH), Luca de Nardis (Uniroma1), Andreas Polydoros (IASA), Valentin Rakovic (UKIM), Daniel Denkovski (UKIM), Mihajlo Pavloski (UKIM), Vladimir Atanasovski (UKIM), Liljana Gavrilovska (UKIM), Paweł Kryszkiewicz (PUT), Adrian Kliks (PUT), Hanna Bogucka (PUT), Bassem Zayen (EURECOM), Raymond Knopp (EURECOM), Erhan Yilmaz (EURECOM), Navid Nikaein (EURECOM), Gianmarco Baldini (JRC)
Participants:	RWTH, Uniroma1, IASA, UKIM, PUT, EURECOM, JRC
Workpackage:	WP7
Security:	Public (PU)
Nature:	Report
Version:	1.0
Total Number of Pages:	99

Executive Summary:

In this deliverable we introduce and analyse the main, relevant Software Defined Radio (SDR) and Cognitive Radio (CR) platforms. Some of the relevant research is also discussed in order to provide perspective for the conclusions. The document aims to provide basic introduction to main available platforms and provide guidance for research groups on selecting between different competing R&D platforms.

Section 1 of the document describes a basic concept of Software Defined Radio (SDR) with relation to Cognitive Radio (CR) as commonly defined in the literature. Following “knobs” and “meters” (metrics) available at different layers of the protocol stack from a wireless cognitive node point of view are presented. Fundamental challenges in SDR design from hardware (front-end) and software point of view, as well as, the latest developments and trends are addressed in the end of this section.

In Section 2 we introduce and analyse the main, relevant SDR and CR platforms. The platforms and testbeds experienced and deployed by ACROPOLIS partners are presented including experimental examples. Other platforms and testbeds are also shortly presented.

Section 3 comprises evaluation and roadmap of SDR/CR platforms, including evaluation of key platforms and roadmaps of the platforms used by various ACROPOLIS partners. Shortcomings and positive experiences on the platforms used and deployed by the project partners are collected in order to share main pros and cons with other research groups to help them on their selection of tools.

Keywords: software defined radio, cognitive radio, evaluation and roadmap

Document Revision History

Version	Date	Author	Summary of main changes
0.1	03.12.2010	RWTH	Initial structure of the document
0.2	15.04.2011	EURECOM	Included EURECOM's first contribution
0.3	06.05.2011	UPRC	Included UPRC's first contribution (3.4)
0.4	12.05.2011	RWTH	Included RWTH's texts (3.2, 3.12, 3.13)
0.4	13.05.2011	KCL	Included KCL's first contribution (Chapter 6)
0.5	13.05.2011	IASA	Included IASA's contribution (Chapter 1)
0.6	13.05.2011	UKIM	Contribution from UKIM added in sections 2.3, 3.1.2, 5.1 and 5.3
0.7	13.05.2011	PUT	Contribution from PUT: Section 3.1 and 3.11
0.8	13.05.2011	EURECOM	Included EURECOM's second contribution
0.5	17.05.2011	UPRC	Contribution from UPRC added in sub-section 6.1.3, update of list of acronyms.
0.9	30.05.2011	EURECOM	Contribution from EURECOM in section 3.9
0.10	01.06.2011	JRC	Contribution to 1.3, 2.4 and 6.
0.11	06.07.2011	RWTH	Editing problems extraction, styles correction
0.12	03.08.2011	RWTH	Sections regrouping, cut of repeated info, adding glossary, …
0.13	09.08.2011	RWTH	CREW federation platform section added, Other Platforms, Testbeds sections updated
0.14	29.08.2011	RWTH	RWTH-edit
0.15	29.08.2011	JRC	Contribution to section 5.
0.16	01.09.2011	EURECOM	Editing EURECOM's contribution and added extra inputs
0.17	02.09.2011	RWTH	Edit
0.18	06.09.2011	EURECOM	Contribution updated: 2.5.4.4 and 3.2.3
0.19	07.09.2011	PUT	Editing and contributions to 2.2, 2.3 and 3.2
0.20	09.09.2011	UKIM	Editing and additional contributions to sections 1.5 and 2.3
0.21	09.09.2011	RWTH	Editing
0.22	26.09.2011	RWTH	Editing and additional contributions
0.23	26.09.2011	RWTH	Coordinator review
0.24	28.09.2011	RWTH	Analysis of platform strengths
1.0	30.09.2011	RWTH	The final major review concluded

Table of Contents

1. Introduction

This deliverable comprises the existing research work in Software Defined Radio (SDR) and Cognitive Radio (CR) domains, describing in detail those SDR/CR platforms or testbeds which the project partners are experienced with. Research challenges, fundamental limitations, good experience and shortcomings of existing SDR/CR platforms/testbeds are presented in this document.

The next two sections of this chapter describe a basic concept of Software Defined Radio with relation to Cognitive Radio as commonly defined in the literature. The following section presents “knobs” and “meters” (metrics) available at different layers of the protocol stack from a wireless cognitive node point of view. The readings of these metrics depend on the internal state of the node, i.e. on the setting of the control mechanisms that influence the performance of the node. When considering the building of whole networks, these are the setting of different protocol parameters or knobs, especially on the PHY and Medium Access Control (MAC) layers. The section encompasses a list of the parameters for different protocol layer, focusing on physical and MAC layers. The last section of this chapter addresses fundamental challenges in SDR design from hardware (front-end) and software point of view, as well as, the latest developments and trends.

The main objective of the second chapter of this deliverable is to describe the existing SDR and CR platforms and testbeds in the context of ACROPOLIS project. Therefore, the platforms and testbeds experienced and deployed by ACROPOLIS partners are presented including experimental examples. Other platforms and testbeds are shortly presented in the second part of this chapter.

Section 3 comprises evaluation and roadmap of SDR/CR platforms, including evaluation of key platforms and roadmaps of the platforms used by various ACROPOLIS partners. Shortcomings and positive experiences on the platforms used and deployed by the project partners are collected in order to share main pros and cons with other research groups to help them on their selection of tools. The last section concludes this document.

1.1 Basic concept of Software Defined Radio (SDR)

The first reconfigurable receivers were already developed during the 1980s for the radio intelligence for the short wave communications. An automatic recognition of the modulation mode of a received signal, or bit stream analysis, was already included in these receivers [104].

A transceiver, being a *Software Radio (SR)*, is characterized by the realization of its communication functions (including all the layer of communication system) as programs running on a suitable processor. Different transmitter/receiver algorithms (based on the same hardware (HD)) are implemented in software. And ideal Software Radio samples the antenna output directly. In a *Software Defined Radio (SDR)* the received signals are sampled after a suitable band selection filter.

A *Cognitive Radio (CR)* is an SDR that can sense an environment, track changes and act. A CR, an autonomous unit in a communication environment, can frequently exchange information with networks it can access, and with other CRs [104]. However, we can question why do we need such autonomous unit?

The electromagnetic radio spectrum, a natural resource, used by transmitters and receivers (transceivers), is licensed by government agencies. However, this recourse is currently underutilized; a large portion of the assigned spectrum is used only sporadically, where geographical variations in the utilization of assigned spectrum ranges from 15% to 85% with a high variance in time [106]. Therefore, A CR offers a way of solving this spectrum underutilization problem. Defining deeper CR features, a CR senses the radio environment with the objective to identify those subbands of the radio spectrum that are underutilized by the primary (i.e. legacy) users and to provide the means for making those bands available for employment by unserviced secondary users [105].The degree of flexibility of a radio platform is of considerable relevance to CR [107]. A commercially viable SDR can be implemented as a mix of Application-Specific Integrated Circuits (ASICs), Field Programmable Gate Arrays (FPGAs), Digital Signal Processors (DSPs) and general-purpose microprocessors. ASIC parameters may be defined in software, however, an ideal software radio has no ASICs, since it implements all the radio's RF conversion, filtering and related functions in software [107]. Figure 1-1 depicts the range of software radio implementations as a function of digital access bandwidth and processor programmability.

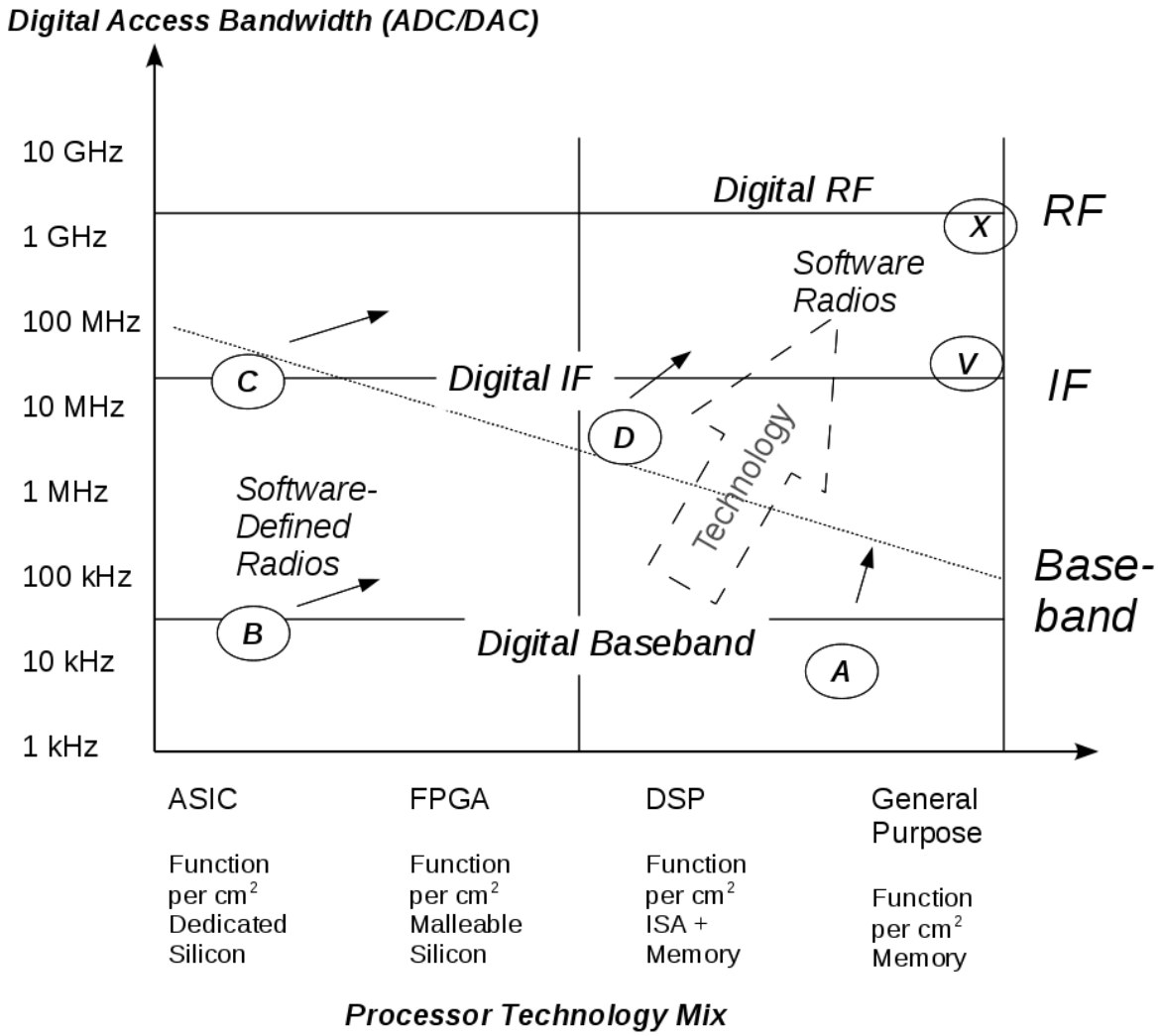


Figure 1-1 Dimensions of software radio implementations (adapted from [107])

The vertical axis stands for the bandwidth at the digital access point, where horizontal represents degree of flexibility, the fraction of functionality that may be changed “in the field” using plug-and-play software. Details can be found in the fundamental work of J.Mitola [107].

Figure 1-2 depicts a *generic architecture* of Cognitive Radio transceiver, including radio front-end and the baseband processing unit.

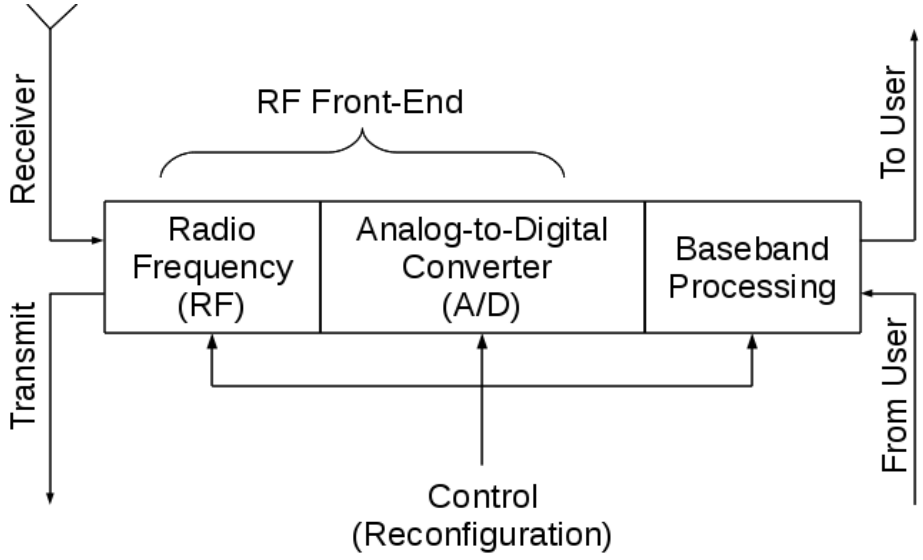


Figure 1-2 Cognitive Radio transceiver (adapted from [106])

Each component can be reconfigured via a control bus in order to adapt to the time-varying RF environment. The received signal is amplified, mixed and A/D converted in the RF front-end. It is modulated/demodulated and then encoded/decoded in the baseband processing unit [106].

In this section aforementioned concepts and definitions are presented shortly based on the well known fundamental references [104][105][106] and [107]. In the following section, a CR radio concept with its relations to SDR is presented from the point of view of a communication-transmission problem, addressing uncertainty from the communication standpoint and the concept of cognition and flexibility in a radio system.

1.2 Basic concept of Cognitive Radio (CR) and its relationship to SDR

1.2.1 Taxonomy of cases

Assume we are interested in a communication-transmission problem (point-to-point, point-to-multipoint, multi-to-multi) where the environment contains unknowns, without the knowledge of which the communication action (transmission-reception) cannot be completed. Since "environment" is typically meant to mean "the channel", this can be viewed as a classical formulation for communicating via uncertain channels. Here, uncertainty may pertain to the physical characteristics of the propagation medium (gain, phase, time-frequency spread, possibly non-linear effects, signal-to-noise ratio, etc.), or interference (besides white noise which is considered) omnipresent, presence of other users (in a networked environment), or of other systems, etc. All communication links are faced with such nuisances, to one degree or another, and a good deal of the signal-design and reception theories address the best way to reign into such uncertainties and reduce their

negative effect. Channel estimation, synchronization, gain control, equalization, noise-level estimation, unknown interference identification/measurement/management/mitigation, co-channel separation, etc., all these are familiar topics that address precisely the unknown effect from the environment that impede the communication action. The acquired knowledge about the value of the *parameters* that summarize this uncertainty will then be gainfully employed either at the receiver for better reception or, possibly also, at the transmitter for a better choice of waveform to be sent (adaptive transmission, cooperative signalling, etc.). There is, therefore, a *parameter-estimation* action that is implied here, and the role of this action is to make the communication task ("mission") better. The system designer usually chooses ways that facilitate such estimation (for example, via the insertion of training symbols, pre-ambles, pilot symbols, etc., or may even choose to do nothing and count on blind processing via, say, non-coherent modulation) at the expense of some channel resource. Such expense is deemed necessary for the overall communication-task execution and it can be optimized for the best trade-off between recourse expense, i.e., utilization reduction one the one hand, and quality of transmission-reception on the other. If the system is *dynamic*, i.e., if the values of these parameters change with time, then the estimating action must adjust itself continuously (tracking). In any case it is clear that, for this set-up, the estimation action[1] is subservient to the communication action, and this can be pictorially represented by the two loops below, where the lower estimation loop (EST) just "serves" the upper communication (COMM) loop. There is nothing that the upper loop does to affect the lower loop. Note that the word "does" pertains to the communication action itself, namely the transmission of the useful (payload) data, not how things are "done" during the design process. So far, then, we have just re-drawn the classical communication landscape in the presence of various nuisance parameters, whose typical engineering description falls either under the stochastic world of "random variables" (or processes), or under the form of "fixed but unknown". Such distinctions, which may reflect either the true state of nature or what we, the designing engineers, choose to model for various reasons (complexity being one of them), has an impact on the eventual design of the communication link and the related performance of the communication action. Obviously, one may also choose to estimate not just the parameters themselves but also their statistics, if unknown. One may build histograms for the desirable densities, for example, then fit parametric models to such histograms and proceed with probabilistic processing (Bayesian, etc.) as if such collected statistics are true. In other words, there can be an increasing set of layers of uncertainty that can be chosen, for higher modelling sophistication and performance; this does not change the essence of Figure 1-3, it just increases the dimensionality of the lower loop.

[1]We call this "estimation action" and avoid the tempting term "learning action" for reasons that should become clear shortly.

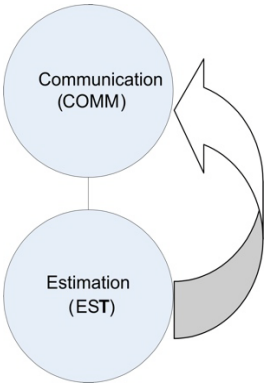


Figure 1-3 Estimation action in the service of the communication task

We can imagine situations, however, where the arrangement of loops is not as per Figure 1-3 but different. For instance, the estimation loop may be what is desired and the communication loop is there to provide the necessary physical action. Although estimation of parameters, in and of itself, does not appear to be a profitable action per se, we can easily think of tasks where this knowledge is very beneficial: position location for handset users and all related services that can be provided to that user as a function of such position is but one example, although currently very popular. There, the picture would look something like:

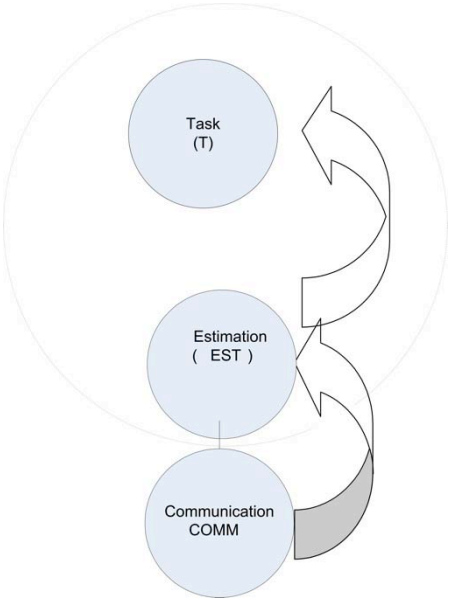


Figure 1-4 Communication aiding estimation, which then aids another task.

Clearly, the upper two loops can be served by designing signals tuned to that task only and not to the communication task. For example, GPS waveforms do not convey communication data, but they are designed just so as to facilitate position estimation (via high-resolution timing estimation, for instance). This, however, falls outside our interest here, where we assume that the COMM loop is always there for some autonomous purpose. So the meaning of Figure 1-4 is that the COMM mission is taking place anyway (for a self-existing reason), but it can affect parameter estimation, which is meant for another task (e.g., position location). So, in Figure 1-4, just as in Figure 1-3, the "effect" arrows flow one way (upwards). Thus, we can simplify this Figure, if we wish, by lumping together the upper two actions and

have only two interacting loops (one being a "super loop"): the COMM loop at the bottom and the EST-T loop at the top, where EST is the above estimation action and "T" stands for any other task that flows from it, for example, the position-location task. This latter task may be connected to another higher task (for instance, a location-based service) but, from our scientific standpoint, this higher connection is of no interest.

What now if arrows loop around? What if the COMM task and the T task affect each other? The situation will then be as follows:

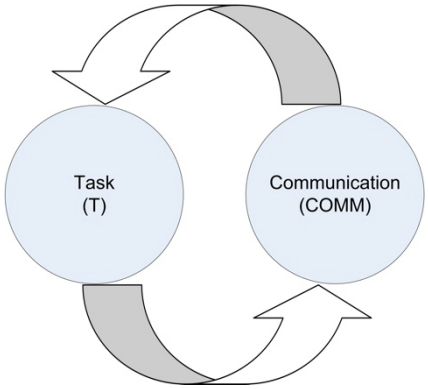


Figure 1-5 Interacting Tasks

Among the many situations imaginable that give rise to this mutual dependence, we can mention two: in the first case, imagine that the Task T is a real-world task whereby the communication nodes move in response to the latest information, as conveyed by the communication task. So if set of first responders is approaching a flaming building, and they communicate via their radio equipment for their moves, then the execution of their main Task is obviously affected by the execution of the communication action, whose quality may in turn also be affected by where the nodes move (due to fading, blockages, etc). In other words, communication may affect physical movement, which in turns affects connectivity and thus communication: mutually-affecting actions.

A second case of interaction is a bit more subtle as it doesn't pertain directly to the physical world but to the mental world: the learning functionality in the field of cognitive radio (CR). *Learning*, in the context of CR, is defined as the process of acquiring new or modifying existing knowledge[2] for the purpose of improving radio-system operation, which knowledge is however obtained by *running the radio system itself* and monitoring its output as a response to specific input stimuli. In other words, for the present definition of learning to hold we assume that this knowledge cannot be acquired by means other than operating the radio system under consideration, i.e., other than acquiring experience by running the system itself. This conforms with the common-sense notion of learning in the living world (trial and error, learning by experience) and also conforms with the standard use of the term in machine learning, system science, pattern recognition, AI, etc. Therefore, it is by definition clear that the learning action is dependent on precisely how the communication task performs under the given stimuli ("trial" stimuli or "real-operation" stimuli; "supervised" or "unsupervised" learning, respectively) necessary for the learning. What is less obvious is how the learning affects the communication operation**. Here enters the science of cognitive radio, whose objective is precisely to explore the ways by which**

[2] We use here Wikipedia definition

learning mechanisms and their accumulated knowledge improve the operation of a communication system. If, for example, we wish to communicate in an interference-polluted environment, the first distinction we should be making is whether this interference is produced by an independent agent or a system, which we don't control, or by a system which is fully or partially under our influence. In the first case, the only thing we can do is estimate the characteristics of that interference ("noise" to us, essentially), and adjust our communication algorithms and protocols accordingly; this would fit the paradigm ofFigure 1-3. If, on the other hand, such interference is produced, at least partially, by the effects of our own system (e.g., the other co-users), it is quite conceivable that we find out certain characteristics of said interference while we operate the system and then adjust the system itself accordingly, for maximum benefit. Here we recognize the distinction between open-loop versus closed-loop (feedback control) systems, except in a more abstract setting, as well as in learning/cognitive terms. The second case clearly implies a dynamic setting of action-reaction-learning-new action, etc., consistent with the cognitive cycle of Mitola [107] but with a more specific delineation of the different actions involved.

1.2.2 Handling uncertainty from the communication standpoint

All three figures above have one thing in common: at least one of the actions is faced with *uncertainty* which is generated by another action or task and thus prevents that original action from acting autonomously. In the first case depicted in Figure 1-3, the communication mission cannot be completed until some parameter values are first estimated and provided (this may be done dynamically or recursively in time, but in any case a future communication step needs prior estimation to be completed first). If this need happen only once (as in a very static setting), then the two tasks eventually decouple and, in the limit, there is no dependence. Whatever had to be estimated was computed "long ago" and that's it. In dynamic (non-static) scenarios, this is obviously not the case; then, the communication action is in constant need of the parameter information-extraction action. How exactly this estimation is performed, how many resources are expended and what the resulting accuracy is, all these comprise popular research themes. The essential goal here is to handle the uncertainty of the unknown parameters in the most efficient way, a goal that is very much dependent on the complexity of the underlying system. A classic way of handling the complexity underlying such uncertainty-handling is the theory of abstraction [8]-[16], whereby the behaviour of complicated communication systems and channels is compactly represented ("compressed") via information-theoretic methods in order to either assess the performance of ever-higher functionalities and layers or to adjust (adapt) transmission to prevailing conditions (e.g., adaptive modulation and coding). If the abstraction models have been pre-formulated in specific parametric form and the only thing missing are the values of parameters pertaining to current reality, then Figure 1-3 is relevant; if the abstraction model itself is modified via the communication "experience", then Figure 1-5 is relevant.

In the second case depicted in Figure 1-4, the ultimate Task (for instance, position location) depends on parameters whose values in turn depend on the actual instantiation of the communication task. So, for example, if we want to locate a cellular base-station tower in space and wish to do so in real time, we must first receive the communication signals that this tower emits, process them to extract relevant parameters (e.g., the time of arrival—TOA), mix this information with other TOA's from other receivers (possibly by exchanging/passing proper *messages*), and then triangulate for the final outcome of the position-location Task. Here, the uncertainty faced by the upper-task geometry engine

pertains to the kind and quality of the set of radio-dependent characteristics that are extracted and mixed. Such characteristics are termed "*features*", using the standard terminology of statistical inference theory. We note here that, whereas in certain tasks such as geo-location (triangulation-lateration) the choice of the proper features to be used is not much in dispute (only their quality and effectiveness is under debate and is typically very much scenario-dependent), the choice of the right feature set for other tasks such as signal sorting and recognition is still a topic of research debate. Although the science of pattern recognition and machine learning provides some initial pointers for extracting features that seem to capture relevant properties of the underlying signal set, the complexity of the task (for large menus) as well as the unpredictability of the medium (fading channels, in particular) make the final proper choice an open research issue. An example of differences in approach here is between the statistical camp (those advocating statistics-related features such as various moments, crossings, properties of samples, etc.) and the probabilistic camp that models reality parametrically as closely as possible and then applies likelihood inference directly, so that the "features" of the latter approach are the likelihoods themselves, or derivative metrics thereof. Similar differences, such as direct (non-parametric) versus indirect (parametric) methodologies can also be encountered in the radio-sensing or radio-field estimation literature. In all such cases, however, the estimation action derives its results from some communication (signal emission) action and then, either directly or via properly derived features, in turn informs the upper (final) Task. Here, therefore, the communication action and its surrounding environment are not the recipients of uncertainty but, rather, the sources of it (uncertainty as viewed by another task).

In the last case of Figure 1-5 (interacting Tasks), the uncertainty that the communication action faces (above and beyond its own environment, channels, noise etc.) is generated by its intimate dependence on another task. If that adjoining task is "learning", then the quality of communication is directly dependent on how well the learning process performed. If, for example, we wish to adapt to a channel, how well the information about that channel was acquired previously, from past communication uses, how well it was compressed and presented, etc., all these have a direct bearing on the quality of the next communication action. In primary/secondary opportunistic spectral access (e.g. IEEE 802.22), there is a fundamental question, from the standpoint of the secondary user, as to whether it can learn the effect of its transmission actions on the primary user: if it cannot (independent systems), all the secondary system does is sense and act (Figure 1-3). If, on the other hand, it can inquire the primary system about the effect of its transmissions (how much interference it is generating), it can facilitate its learning mechanism and improve its actions (Figure 1-5). In *ad hoc* radio networks, radio connectivity can be affected by physical movement of nodes, as long as quality of communication can be learned by information sharing (Figure 1-5).

A proper combination of the above figures can provide an interpretation for the recently popular notion of Radio Environmental Maps (REM), depending on the scenario at hand. Assume we wish to communicate in a particular slice of time, space and spectrum, based on information about the prevailing interference conditions. This interference is generated by some radio sources. If we wish to approach the interference-field-estimation task (i.e., one entry to the REM) parametrically, the incorporation of the explicit communication action of the interference-generating sources leads us to Figure 1-4: we collect information (sensing) from the action of the signal(s) emissions, we proceed to estimate parameters and then feed the final task of field estimation. Once this result is uploaded into the REM, the

functionality of Figure 1-4 is complete. Then, depending on the situation, we either proceed as per Figure 1-3 or Figure 1-5: if the result of the estimation action of the REM affects the communication task of a *secondary* system, namely one that is different from the one generating the interference (usually referred to as the *primary* system), then the situation, as far as the latter (secondary) system is concerned, is as per Figure 1-3. If, however, the actions of a (single) communication system affect the measured emissions and count as interference to be sensed and input into the REM, then the REM and the COMM action are related as per Figure 1-5. In essence, this situation can be combined into a new Figure 1-4**.(a),** by adding a downward arrow from the Task (the REM formation) down to the COMM action, the two being closely interwoven in a larger loop.

To sum it up, **we have created here the notion of dependent (interacting) actions in order to introduce a unified framework, which allows for a conceptual handling of the role of information uncertainty as it affects some communication action (in single-link systems, in networked systems, etc.)** There are, of course, multiple and layered ways to deal with uncertainty in life and the same applies in communication science. We can list a few starting from the simplest: ignore it. To ignore uncertainty, i.e., to act in *agnostic* ways with respect to its presence and effect, one has to design systems whose operation and performance do not depend on said uncertainty. Another way to phrase it is to say that the design must be fully robust with respect to the different faces of that uncertainty. A classic example is non-coherent communication systems: it is known in advance that the channel will introduce unknown phase rotations during transmission, and the designer can either opt to estimate such rotation at the receiver (coherent communication) or may opt for a con-coherent (energy-based) signal set. Although it is known that non-coherent communication is usually inferior in performance to a coherent one, the ability to ignore such phase uncertainty during operation is an appealing feature in many cases. Another example comes from *ad hoc*, multi-hop mobile networks (MANETs): classic designs pre-suppose a warm-up phase of "learning" neighbourhood information for each radio node in the network, on the basis of which routing tables are formed and messages are subsequently routed. There exist, however, alternative designs (Opportunistic Large Arrays (OLA), Barrage-Relay Networks (BRN)) which are purely agnostic with respect to such neighbourhood information, and thus do not rely on routing tables that must first reduce such topological uncertainty (the training period) before proceeding with the communication action. Agnostic designs, in general, tend to be very robust to the relevant uncertainties (their reason for existence!) but do offer inferior performance in environments that can be learned well without major cost or resource expenditure.

A step up in sophistication is not to be agnostic, but to, in effect, deal within uncertainty by forming one's own "reality", one's own estimable version of the surrounding environment; this is what the previous section elaborated upon. This is typically performed by assuming a model in parametric form and then using various procedures to learn (estimate) the value of the pertinent parameters. Once this is achieved, the model is ready to be used for the communication action. The engineering challenge here is to design models that simultaneously capture the true behaviour of the uncertain elements, yet achieve this with a minimum of parameters (thus comforting to the principle of Occam's razor and the related MDL principle [17]). In many practical radio environments (e.g., cellular systems), fairly precise parametric modelling is feasible, based on a mixture of physical principles and past experience. In those cases, the design incorporates parameter-estimation procedures (training, etc.) for completing the model knowledge and for the communication action to

proceed. In other, more challenging cases (for instance, surroundings with unknown or varying radio-propagation laws), the exact behaviour of the uncertain system is hard to capture with precision, and then one resorts to compressed (abstracted) models, with fewer parameters, upon the estimation of which a parsimonious yet sufficient representation or true reality is obtained. In other words, in these latter cases, on grounds of reduced-complexity representation one deliberately does not attempt to eliminate modelling uncertainty completely but does manage to build a sufficiently good working system.

1.2.3 Cognition and Flexibility in Radio Systems

The above discussion placed, among other things, the notion of *learning* in the context of communication (in general, and radio in particular) systems as a means for handling information uncertainty. Furthermore, it put it in a framework of dependent actions, restricted to the case where one of the actions is the communication task. *Cognitive radio* [18], a popular research and development topic of the day, can be viewed as a system where learning leads to a subsequent meaningful (i.e., optimized) action which is relevant to the radio usage and which exploits the knowledge accumulated during the learning process. In that sense, learning is a necessary first part of the execution of the cognition cycle that must be satisfied for the terminology to apply.

What are the properties and attributes that the radio system itself must satisfy in order to qualify for usage in such a cognitive framework? In particular, what are the requirements on *flexibility* that must be satisfied in order for the radio to qualify for use in any CR scenario? To be specific, flexibility in this context is interpreted as the ability to change (morph) according to external stimuli or demands [21]. To understand the ramifications of this interpretation, however, we need to distinguish between *system-centric* radio flexibility and *platform-centric* radio flexibility. System-centric flexibility addresses topics where the radio-*system* flexibility is the sought goal, and where the flexibility (or reconfigurability) of the implementation platform can be abstracted to a sufficient degree without need for detailed description of the innards of the specific execution machine. The emphasis of this domain is on adaptation and optimization methods that typically pre-suppose the existence of multi-modal/multi-standard Radio-Access Technologies (RAT) or, in any case, other mechanisms for affecting the lower layers of the system according to higher-layer decisions and directives, as well as on how this available flexibility at all radio levels/layers can be exploited towards improving various physical and network-level cost functions and performance metrics. Vertical handoff (VO) protocols is a classic such example of a suite of solutions that rely on the simultaneous existence of multiple RAT's serving the same area and which, by pre-supposing the existence and flexible operation of multi-standard terminals (but without undue consideration of their implementation details), tries to satisfy various metrics as total capacity, fairness between users, etc.

On the other hand, *platform-centric* radio flexibility, which can also be termed "*equipment-centric*", has an obvious conceptual anchoring on the computing, software (SW) and hardware (HW) fabric that executes the designed flexible functionalities. It addresses modules, components, "nuclei", sub-systems (like flexible RF front-end for wideband operations), baseband HW/SW partitioning, multi-core structures and System-on-Chip (SoC) concepts as well as the related algorithmic, protocol and programming architectures that affect (and are affected by) flexibility requirements and specifications. Anything for which the nature, design and form of the underlying execution platform plays a significant role (regardless of membership in the classic layer or protocol stack) belongs here, with obvious emphasis on the multi-disciplinary and multi-layered nature of the design challenge addressed. Terms like "reprogrammable HW/SW fabric", "reconfigurable computing resources and logic",

"Flexible/Adaptive/Reconfigurable (FAR) modules[3], algorithms and design tools", "*real-time* (on-the-fly) reconfiguration" versus "*design/deployment-time* reconfiguration", and so on, find their natural home in this category of radio flexibility.

With this understanding, we can now re-examine the three Figures above from the standpoint of flexibility requirements for the radio system. It is immediately clear that the situation described in Figure 1-4 does not imply any such requirement, since the communication task is solely the source of signals but not the recipient of any Task-related input. It operates independently and is not affected by any upper-task conclusions. On the other hand, in Figure 1-3, it is clear that the communication system must be able to accept the fresh inputs of the parameter values produced during the estimating action and re-compute certain aspects accordingly. Therefore, a minimum of re-processing capability is necessary for this to happen, meaning that the radio is at least adaptive (and, thus, flexible, at least in this sense of the term). We should point out, however, that adaptive signal processing is by now an established practice and it does not necessarily qualify as a very advanced feature. Practically all modern systems re-estimate periodically the channel and adjust the coefficients of their reception accordingly (e.g., adaptive equalization), an omni-present practice for all cellular systems from 2G and on. Therefore, a rudimentary version of flexibility at the reception side is already commonplace, and current interest has already shifted on adaptation on the transmission side, for reasons of maximal resource usage, interference management, etc. Transmission-based adaptation is flexibility of conceptually higher order; it is in fact "proactive" adaptivity (as opposed to "reactive" adaptivity at the receiver side). Such proactivity can lead to significantly higher efficiencies for the total system but also requires higher sophistication at the design stage, as it essentially requires a pre-assessment of the possible "system modes" that the system may face in operation. All these modes (or a meaningful subset thereof, sufficient to cover the "system space" without major losses) need to be pre-calculated and pre-inserted in the system for use in real-time, which implies a sophisticated design effort. In addition, we may envision cases where the mode set itself gets enriched in real-time, based on recent information and experience (with the involvement of the reception side), in which case we are faced with a classic case of learning and adjustment, as described by Figure 1-5.

The very structure of Figure 1-5 implies an interactive adjustment of the communication task as a function of the results produced during the execution of the broadly defined other Task; therefore, flexibility is an obvious requirement for the radio system under consideration. Furthermore, since the other Task may include learning dimensions, thus requiring cognitive functionalities for the composite system (jointly executed Tasks), it is clear that Figure 1-5 forms the basis for cognitive-radio design and operation.

1.3 Knobs and Meters

A state of a wireless cognitive node can be described using various metrics, Key Performance Indicators (KPIs), or attributes collected at different layers of the protocol stack. The readings of these metrics depend on the internal state of the node, i.e. on the setting of the control mechanisms that influence the performance of the node. In case of networking these are the setting of different protocol parameters or knobs, especially on the PHY and MAC layers. Below we provide a list of the parameters for different protocol

[3]Flexible changes that tend to be viewed as small or gradual tend to come under the term "adaptation", whereas larger changes of structural nature tend to be called "reconfiguration". The notions of adaptivity and reconfigurability imply designs that endow the respective abilities to perform such changes.

layer, focusing on physical and MAC layers. (For the MAC solutions we consider mostly CSMA/CA-based protocols).

1.3.1 Knobs

1.3.1.1 Physical layer + Radio: Knobs

Multiple Radio Interfaces: It is a common practice for modern devices to have several wireless interfaces, several of which can be active simultaneously. A cognitive wireless node should be able to support several wireless interfaces and efficiently utilize them, including the functionality of splitting the traffic between the interfaces, based on both technical considerations of the users and device constraints, such requirements from currently active data flows or amount of battery resources, as well as non-technical one, such as economical concerns.

Transmission power: Various protocols and algorithms require controlling the transmission radius of a sensor node, which may be achieved though the radio transmission power. This parameter can be used by applications to effect the energy consumption at a node, capacity of the network etc.

Radio channel/width, operational frequency range: Often radio transceivers support multiple channels and it is possible to operate in a specific channel to avoid interference from other devices. It is also possible for some software-defined radios to operate on variable bandwidth (channel width), thus creating opportunities for more efficient spectrum usage.

Modulation scheme: Often radio transceivers support several kinds of modulation schemes and it may be exploited accordingly to achieve the desired receiver sensitivity and data rate requirements by the applications.

Power modes of the radio: It is desirable for software-define radios to support low-power modes the modern radio transceivers. For example, wireless sensor nodes support different low-power modes and exposing these modes for higher layers and applications allow executing various types of energy efficient schemes. An added control over this feature also enables the nodes to switch the radio in a suitable mode having more application specific information.

Number of subcarriers, subcarriers used: Some physical layer technologies, like OFDM, use multiple subcarriers for data transmission. The number and choice of the subcarriers to be used can be controlled by a cognitive radio to more efficiently mitigate the external interference, for e.g. avoid using a narrow band occupied by a primary user without vacating a whole transmission band.

Other possible metrics include: checksum methods, range and precision of random generators, forward error correction schemes, list of active interrupts (header detection, CRC failure, preamble detection, frame reception, etc.)

1.3.1.2 MAC/Link layer: Knobs

Persistency checks: Certain MAC protocols can attempt data-transmission based on the persistency check. Persistency value affects the overall throughput and fairness. Therefore, this knob can be useful to be exposed to the control layer of the cognitive node.

Channel activity detection / Clear Channel Assessment (CCA) threshold and interval: The CCA duration and CCA threshold enable a CSMA based MAC protocols to judge the channel

activity and hence attempting transmission or refraining from it. A low threshold and small interval may lead to false channel assessment and too long durations will waste energy. The CCA threshold should be adjusted according to the noise floor of the wireless spectrum. An application will be able to control the threshold of the CCA to manipulate the channel activity detection of the CSMA MAC protocol.

Sleep duty cycle: For low data rates, control traffic, it might be useful for a node to periodically go to sleep and active state (exercise a duty cycle scheme) to save energy. The duration of sleep interval (possibly as a percentage) may be exposed to the control plane so that the trade-off between latency and energy consumption KPIs can be achieved.

Control data handshake: Example of the control data handshake is the Ready-to-Send (RTS) and Clear-to-Send (CTS) handshake is use in CSMA/CA-based protocols to mitigate the hidden terminal problem. Depending on the type of the transport protocol, ability to detect the hidden terminal problem the cognitive node might decide on using this handshake mechanism.

Initial/Maximum backoff interval, backoff mechanism: The parameters of the backoff window mechanism (initial and maximum values), as well as the function according the back window is increased is one of the core components of the CSMA MAC protocols. If this feature is exposed to the control plane, the contending ability of the nodes for the same channel can be effected and hence the overall latency and fairness metrics.

ACK enabled, Retry limit: Acknowledgments for the received data frames, as well as the number of times a data frame will be tried to be sent before permanent dropping affect both the latency and the reliability of the link. Settings of these knobs depend on the application requirements and the link quality. For example if a video streaming application is good at compensating for the lost frames, but cannot handle high delays them the high retry limit might just have a negative impact on this application performance. Otherwise, typically the ACK with low retry limit typically suffices for most TCP-based applications with moderately good quality of the link.

1.3.1.3 Network layer: Knobs

Queuing algorithms and their metrics: Queues and buffers allow controlling different data flows according to multitude of QoS requirements. These knobs exist on different protocol layers, however, the network layer is the traditional layer where these parameters were exposed as part of the routing functionality. One can distinguish, like it is done in Linux, for example, between three mechanisms how the queuing can be controlled. These are Queuing Disciplines, like Token Bucket Flow (TBF) or Random Early Detection (RED), Class-Based Queuing that can be applied on top of the disciplines, and various Filters, Polices and Classifiers. Additionally one can control the sizes of the respective buffers.

Routing protocols and their parameters: In case of wireless multi-hop networks the choice of the routing protocol and its settings heavily influence the achieved performance. The classical example is the use of special ad-hoc protocols, like AODV [19], in case of highly mobile and dynamic environments.

1.3.1.4 Transport Layer

Protocol (e.g. Connection-oriented/connectionless) and their flavours: The nature of the transport protocol and internal control mechanisms influence heavily on the throughput, latency and the packet loss rates achieved by an application. Two distinct well-known

examples are TCP and UDP transport protocols. Typically the control plane cannot directly influence on the type of the transport application an application will choose, however potentially this knob can be useful to further match the quality of the underlying wireless media to the transport mechanisms applied on the upper layers. However some of the knobs of the transport protocols can be set through the operating system. Examples are flavours (versions) of the protocols, like TCP flavour Veno, Reno, etc.[20], and datagram sizes.

1.3.1.5 Application Layer (some examples)

Data rates, Data size/Quality, Codec: Depending on the environmental conditions cognitive nodes may limit the applications from transmission of high quality services to ensure that user are still able to receive the service they require. Application may adapt the quality of their services dynamically by, for example, adjusting the data rate, the size of the file to be downloaded, or use adaptive codecs.

1.3.1.6 Shared between multiple layers:

Update intervals/ Timeouts: Many protocols use timeout or update intervals to match the dynamics of the network topology or the changes in the cumulative link quality. There is also a need for localization protocols or time synchronization protocols need to update the positioning information, time-skew from time to time. Information of the update intervals as the ability of the control logic of a cognitive node to set these intervals or propagate to the individual protocols information of the boundary conditions for these intervals will enable more efficient exploitation of the wireless media and increase the network capacity.

Queue/Buffer control mechanisms and sizes: These knobs are applicable to any protocol layer and protocol that can accumulate forwarded and scheduled packets instead of just dropping them, when the direct sending is not possible. Depending on the desired complexity level different parameters of these knobs can be set.

Segment/Packet sizes: Fragmentation of the packets in the TCP/IP stack can be done on the network (IP) and MAC layers. Additionally many applications have a direct control on the size of the data they want to send. The size of the data frame send of the wireless media has a direct effect on the frame's delivery probability. Additional known issue is the upper-layer re-fragmentation occurring if the maximum data frame size on the lower layers is smaller than on the upper layers. This process can cause additional delays and even packet losses in a network.

QoS settings: There exist different mechanisms to propagate Quality-of-Experience (QoE) requirements and their mappings to Quality-of-Service (QoS) related knobs. These can be done, for example, through prioritizing of the data flows (e.g. though queuing control mechanisms and Type-Of-Service (ToS) field in IPv4 header), deployment of specific protocols like DiffServ [22], or complying with specific wireless standards like 802.11e [23].

1.3.2 Metrics

1.3.2.1 Shared between MAC, Network, Transport and Application Layers

Latency: Packet delivery **l**atency from the source to destination is a metric that can be computed at different protocol layers and may include both a delay over a single hop communication and an end-to-end delay over several hops. This metric is useful to decide

which links should be used depending on the type of the data and its urgency. Exposing this feature is important for the timeliness requirements of the targeted applications.

Error rate/reliability/quality of the connection: The reliability of the connection (link quality over one hop, or end-to-end reliability) and its quality may help a cognitive node in employing various types of error control schemes at different protocol layers or doing other corrective actions in order to enhance the overall network performance. Depending upon the reliability of the link, the cognitive radio may decide to adjust the traffic through the link, make routing decisions or make DSA decisions.

Packet drop rate and bit error rate: Packet drop rate monitored at different layers may enable the cognitive radio to see the reliability of the link. A packet drop ratio of 100% indicates that the link is no more available and new links are needed to be found. With higher packet drop rates, the node may try to find alternate routing schemes and adjust traffic through the link if possible. This in turn will save energy and reduce latency associated with retransmission of packets.

Number of packets delivered (conditioning on the packet type can be included): This metric is tightly coupled with the packet drop rate and throughput metrics. We discuss this metric separately, as it provides important insights, e.g., into the overhead imposed on the network by different protocols or the structure of the control traffics, performance of individual data flows.

Throughput (goodput): Amount of bytes delivered per time unit. This quality metric is used at different protocol layers to characterize the performance efficiency achieved by a single protocol or a whole stack. Care should be taken in exploiting this widely accepted metric, as it along cannot provide insight in the application satisfaction from the achieved throughput (e.g. too many packet losses), as well as the overall network fairness aspect.

Intermittent Connectivity: With successful packet received ratio, the connectivity characteristics may be determined and links with intermittent connectivity or no connectivity can be identified. The cognitive node will be able to avoid these links in an intelligent manner to ensure high reliability. This will also result in low latency and an overall less energy wastage by eluding retransmissions associated with unreliable links.

QoE metrics: There exist several ways how QoE requirements defined by network stakeholders can be mapped to QoS metrics and other KPIs exposed by the protocol stack. We discuss some of the approaches, including the utility-based mapping.

Fairness: This metric allows monitoring how effectively the network resources are shared between nodes or individual data flows. Fairness can be established for various KPIs, such as throughput, latency, channel access probability, etc. One of the common ways to calculate fairness according to Jain's index [24][25].

1.3.2.2 PHY layer

Interference characterization (spatial and temporal): The information on the level of interference, its spatial and temporal structure may help cognitive radios to decide on the transmitting frequencies, as well as on particular PHY and MAC settings.

Received Signal Strength: The received signal strength of the radio packet can indicate the reliability in the link and also hint the physical proximity of the other node.

1.3.2.3 Network Layer

Route life-time: This feature indicates the time when the route was established. Depending on this feature, a cognitive node can decide to update the route. In case of high mobility or if the route has not been used for very long time, a cognitive node may demand to establish new (update) routes.

Number of hops: Number of hops from the source node to the destination node will enable a cognitive node to assess the latency, the proximity of the destination from source and very roughly the energy requirements in using the path. Using this feature, it can have an insight into the vulnerability, latency and energy consumption in exercising the route. Therefore, different topology control schemes can be applied also by using this information.

1.3.2.4 Transport layer

Level of congestion: The transport protocols may maintain a state for the level of congestion seen in the network. This may be exposed to the control plane so that the applications may exercise intelligent mechanisms to control the amount of traffic depending upon the levels of congestion.

1.3.2.5 Security

Types of encryption (algorithm, key length etc.): Different types of encryption schemes may be applied to ensure the desired level of security in the data transport. An interface to various types of schemes and their parameters may be exposed to the network protocols. The cognitive node may decide of the type of the encryption scheme required depending upon the data traffic.

Types of authentication: Various types of the supported authentication methods may be chosen depending upon the type of service offered by the networking protocols. Since applications are very well aware of the types of authentication suited for various services, it is very logical to expose this feature in order for the applications themselves or the cognitive functionality to determine the type of authentication required.

1.3.2.6 Topology

Neighbourhood: Many of the topology control schemes maintain the number of neighbours of each node. This information may be used to exercise topology/neighbourhood based energy efficient schemes and to share traffic on the network in an optimal fashion.

Existence of (a)symmetrical link: Due to the radio propagation effects, certain links may be asymmetric in nature. Routing protocols may also decide to use the links in an asymmetric fashion depending upon the radio links. Information of the asymmetricity in the links enables the cognitive node to estimate the latency and the energy cost in using/exercising the links.

Mobility: The level of mobility in the network may help applications to choose the positioning update intervals accordingly. If the network is static, it is not desirable to get the new position updates of the nodes after short intervals from time to time. Exposing this feature can also help to update various kinds of tables e.g. routing, neighbourhood, etc. and to exercise all the mobility aware application features.

Density of nodes: The number of nodes in a certain area can help the cognitive node to exercise various types of topology control, clustering, data aggregation, network data processing schemes, etc. in a better way.

Position/localization: Positioning information is very useful to the cognitive node, as it allows exercising various collaborative schemes, predicting topology changes and employing other optimization possibilities.

Number of active nodes: The number of active nodes in a certain region can be controlled using various cluster formation schemes. The nodes may switch their states from active to sleep and vice versa to prolong the overall life-time of the network. The cognitive nodes may control the desired levels of connectivity, latency and throughput as demanded by the application requirements by exposing this feature.

1.3.2.7 Other

Synchronization error: All the network time synchronization schemes develop error that accumulates with time. A feature such as time synchronization error at a particular node with respect to a reference node helps the system to update the time-synchronization. Since different control mechanisms have different requirements for the time synchronization, the magnitude of error indicate the node when an update is required for the particular application case.

Remaining battery level -- Hardware: Battery level monitoring can help to optimize the usage of the resources of a node in a way to increase the life-time of the network. Nodes with less battery level may be used less heavily to avoid complete drainage of the battery.

Table 1-1 Example list of parameters passed from MAC to PHY.

Functionality/Interface	Parameter	Possible Value/Range/Unit
Timer	Precision	Second, microsecond, millisecond.
Timer	Type	Oneshot, periodic
Timer	Length	Any
Carrier Sensing	Duration	0.5ms – 10ms
Carrier Sensing	Threshold	dBm
Checksum	Length	8bit, 16bit, 32bit
Checksum	Polynomial (polynomials can be expressed also as a binary /hex)	$x^4 + x + 1$ = 10011b =0x13
Random Number Generator	Range	(0,1), (0,10)
Random Number Generator	Precision	0.01, 0.1, 1,
Radio Control	State	Receive, Transmit, Sleep.
FrequencySelection	Frequency	Hz
Transmission Power	Power	dBm

Table 1-2 Parameters required from PHY for MAC

Functionality/Interface	Parameter	Possible Value/Range/Unit
Timer	Status	Running, stopped
Carrier Sensing	RSSI	dBm
FrequencySelection	Channel Status	Busy, free
Radio Control	State	Receive, Transmit, Sleep
FrequencySelection	Frequency	Hz
Transmission Power	Power	dBm

1.4 Challenges in SDR design

The continuous growth and evolvement of the communications industry yields easy, rapid and cost-effective modifications of radio devices in order to support the new and emerging technologies. SDR is a promising technology that does not require new hardware for supporting a plethora of various physical layer functions (instead, they are being software defined). However, there are still many challenges, both in the hardware and the software segment of the USRP2, that need to be efficiently tackled prior to real-world SDR deployment. This subsection shortly revisits the most prominent ones.

1.4.1 Hardware: Computing Platform

One of the fundamental challenges for the deployment of SDR technology is to provide the necessary computational capacity to process the waveform applications, in particular the complex and high data rate waveforms and especially for units with strict power- and size limitations. This is particularly challenging for small handheld units or multi mode terminals. The power consumption must be below certain threshold to minimize the battery discharge time. Important components in the computing platform are the Digital Signal Processor (DSP) and Field Programmable Gate Array (FPGA).It is quite likely that DSP and FPGA may be increasingly used in SDR to support the needed levels of flexibility and reconfigurability. This flexibility and reconfigurability come at a cost, as dedicated Application Specific Integrated Circuits (ASICS) are always cheaper than DSP and FPGA. Therefore, on average, every 18 months we can expect a doubling in processing power for the same volume, power consumption, and cost. This equates to an order of magnitude (or 10X) improvement every six to seven years. FPGAs offer re-programmability and the simple advantage of high levels of parallelism that cannot be achieved by the essentially sequential DSP. Parallelism is an important feature as many algorithms used in wireless communications it requires parallel processing. FPGA presents the disadvantage of a significant power penalty, especially in the static power consumption. Unfortunately, this problem is not likely to disappear as FPGA devices move toward smaller transistor geometry to achieve higher chip density and faster dynamic speed, the leakage current in each transistor goes up substantially. This is an important issue for handheld terminals but not for base stations.

1.4.2 Hardware: Front-End

The main challenge is to design a front-end, which can efficiently support transmission and reception in a wide range of frequency bands. Beyond the technical challenges, there are

also economical challenges, as the resulting product should not exceed the cost thresholds defined in the commercial market.

The front-end is composed by the following components:

- **Antennas:** a SDR may support a wide range of air-interfaces in (theoretically) a wide range of frequencies, but the performance can decrease substantially. Ideally, antennas have to be impendence or frequency invariant, which can increase the price of the base station or terminal. A trade-off is the availability of invariant wideband antenna against the cost of the antenna and the capability to provide a constant frequency and impedance response. Innovative type of antenna like fractal antenna and software-controlled micro-electromechanical devices may facilitate the requisite gains in efficiency.

- **Amplifiers:** to achieve all the benefits of SDR technology, SDR communications systems must be equipped with high power amplifiers. Specific urban environment such as buildings can introduce at least 30 dB propagation loss. For SDR technology, amplifiers must provide limited distortion for a wide range of frequencies. These technical requirements are usually in conflict. It is possible that only few parameters can be optimized and only for specific platforms: base stations do not have stringent requirements in terms of battery power and weight.

- **ADC/DAC** represents another important trade-off. ADC/DAC with higher sampling rates and bits imply a higher processing power of the other data acquisition devices (e.g., DSP/FPGA) and therefore increased power consumption. On the other side, higher samples rate do improve the noise spectral density (NSD). In an ideal software radio, ADC/DAC must be as near as possible to the antenna. On the basis of the needed dynamic range and the requested wide frequency bands, the current ADC/DAC technology is not able to implement an ideal software radio, especially for mobile device. Digitizing at the antenna is not a realistic design decision as there is pre-selection and down-conversion to Intermediate Frequency (IF). This means that analogue pre-selection components may be present.

- **RF Filter and mixer.** True SDR systems include software reconfigurability up to the power amplifier; this means that RF filters and mixers are also reconfigurable, something not available today at the costs required by the commercial market.

1.4.3 Software

In this section, software includes the real-time operating system and the software framework, which is used to support the activation and execution of software waveforms. The current *Software Communications Architecture (SCA)* framework and *Common Object Request Broker Architecture (CORBA)* middleware was designed for the military requirements, which are quite different from commercial requirements. SCA is considered to be very resource intensive and it does not fit in the business model of the commercial market, where cost effectiveness is a primary requirement. The current evolution of SCA NEXT can provide a higher degree of flexibility and efficiency. In SCA NEXT, the middleware does not need to be CORBA but it can be Remote Procedure Call (RPC) as in the Android architecture.

Other commercial models could be adopted for the development of commercial SDR. IEEE 802.21 Media Independent Handover or Mobile Industry Processor Interface (MIPI) Alliance are examples of industrial standards, which can be reapplied to the SDR technology.

The definition of a software framework to support the portability of waveforms across different SDR platforms is the main enabler for the definition of the "horizontal model" of SDR market, where SDR HW manufacturers and SDR SW developers are able to produce independently their artefacts.

In a "vertical model" a single manufacturer defines both the SDR HW platform and SDR SW waveform. Until now, the SDR market is mostly vertical. In a vertical model, the manufacturer can optimize the SW for a specific HW, but the concept of SW portability cannot be implemented.

The main challenge in this area is to define a complete software framework which can support the "horizontal model" and waveform portability but which is also highly performing in comparison to SDR based on the vertical model.

1.5 Latest developments and trends

Software Defined Radio and Cognitive Radio technologies have received increasing attention from industry, research and regulatory entities. The concept of SDR has evolved from of Joseph Mitola's seminal work in [1] in 1992, where he presented the shift from digital radio to multiband multimode software-defined radios where most of the functionality is provided in software, rather than hardware like in the conventional wireless communication systems. Over this period of time, many different definitions of SDR have been presented by different entities. The Wireless Innovation Forum (previously called Software Defined Radio forum) has defined the following "tiers" for Software Defined Radio:

- Tier 0 – Hardware Radio, which is a baseline radio with fixed functionality.
- Tier 1 – Software-Controlled Radio, where the radio's signal path is implemented using application specific hardware.
- Tier 2 – Software Defined Radio, where most of the radio functions are performed in software. For example, the signal path can be reconfigured in software without requiring hardware modifications.
- Tier 3 – Ideal Software Radio, where software programmability extends to entire system.
- Tier 4 – Ultimate Software Radio, which has full programmability, may operate in a broad range of frequencies and can switch from one air interface/application to another in a limited time (e.g. milliseconds).

While, Tier 4 Ultimate Software Radio may still be difficult to implement at reasonable costs in a short timeframe, Tier 1 and Tier 2 Software Defined Radio are already available in the market for narrowband communications. One example is the Liberty terminal by Thales, which supports analogue Professional Mobile Radio and APCO 25 communications for Public Safety.

For a number of years the focus of SDR research was on military applications. The JTRS (Joint Tactical Radio Systems) program [2] and [3] is intended to permit the Military Services to operate together in a "seamless" manner via wireless voice, video, and data

communications through all levels of command, including direct access to near real-time information from airborne and battlefield sensors. JTRS is envisioned to function more like a computer than a conventional radio and is to be upgraded and modified to operate with other communications systems by the addition of software as opposed to redesigning hardware - a more costly and time-consuming process. A single JTRS radio with multiple waveforms can replace many separate radios, simplifying maintenance. The additional advantage is that because JTRS is "software programmable", they will also provide a longer functional life. Both features can offer potential long-term cost savings to the military organizations.

The JTRS is built on the SCA, which is a software framework created to allow that tells designers to build component-based SDR applications and waveforms. The components are SW processing modules with input and output ports, context dependencies in the form of processing element dependencies, and with settable properties. The logical devices are abstractions for HW modules (also called Hardware Abstraction Layer or HAL) that provide basic functions like the processing of the information stream. The SCA is a distributed systems architecture, allowing the various parts of applications to run on different processing elements, i.e. each component is deployed on one of a set of available processors. The communication between the components, and between components and devices, is based on the CORBA middleware. For communication with specialized processors, e.g. DSPs or FPGAs, SCA advises the use of adapters between CORBA and these units.

The goals of the SCA are to:

a) Provide portability of waveforms between different SDR platforms. Ideally, a waveform could be implemented on a SDR platform and then moved or rebuilt on another one without having to change or rewrite the waveform.
b) Reduce development time of new waveforms through the ability to reuse design modules.
c) Build on evolving commercial frameworks and architectures.

To achieve these objectives, the SCA provides the following components and functions:

- Radio management functions,
- Domain Manager,
- Application Factories,
- Applications,
- Device Managers and Devices

While, these components and functions are essential to support a variety of waveform, they still do not enable fully portability of the waveforms. Significant pieces that are not standardized by the SCA itself are the APIs to the services and devices of the system platform. Since these are linked to the actual implementation of the system platform, they are supposed to be standardized per system or domain, as is clearly pointed out in the SCA 2.2.2 specification.

A recent SCA evolution is SCA NEXT, which aims to design a more flexible and efficient version of the SCA framework.

SCA NEXT will provide the following benefits in comparison to SCA:

1. "SCA Next" is more scalable, lightweight, and flexible than SCA 2.2.2. It is compatible with different radio sizes ranging from handheld, vehicular or base stations.
2. SCA Next incorporates advances in portability for DSP and FPGA processors and new design patterns for its Application Program Interfaces (APIs).
3. CORBA is no longer required, allowing lightweight radio-specific middleware (like the Android's RPC) for communication between software components and hardware devices.
4. Registration of components and devices has been redesigned, incorporating a 'push' model that substantially reduces communication. This enhancement facilitates dynamic or static configurations and reduces start-up times.
5. A flexible specification for the Application Environment Profile (AEP) defines the minimum operating system features required for a specific radio platform.

In all these activities, the JTRS has collaborated with the Wireless Innovation Forum (www.wirelessinnovation.org) or WinF. The WinF has produced many reports in different areas: from market studies to SCA specifications, to security aspects and the application of SDR in the Public Safety domain.

The *European Software Radio Architecture (ESRA)* is an ongoing standardization activity at European level. The goals are to ensure waveform portability and SDR reconfigurability. The ESRA standardization activity will be implemented through existing projects at European level like ESSOR, WINTSEC [4] and its follow-up EULER [5], which is focused on the application of SDR for improved joint interoperability in Public Safety and defence. The *WINTSEC project* has laid the foundations of ESRA, which however has not yet reached the level of a standard but rather an architectural framework; the items defined in the ESRA document are not actual requirements but mere recommendations. The ongoing *EULER project* is expected to provide further ESRA recommendations extensions. Many organizations and industries in Europe are involved in this process. The European Defence Agency (EDA) is a main player in this process.

As described above, SDR technology has received considerable attention in recent years in the defence domain and (in smaller part) in the public safety domain.

The commercial market was not initially investigating SDR technology with the same attention due to the perception that SDR equipment (especially handheld) can be too expensive for the mass market, where interoperability issues are not very relevant and non-recurring costs of ASIC designs are minor in terms of the huge volume of commercialized terminals (in the order of billions of units). Nevertheless, the decreasing costs in integrated circuits and FPGA and their increasing processing power have driven many companies and organizations to investigate further in SDR technology for commercial market. Strong examples are VANU products (http://www.vanu.com/) and the Uni-RAN SDR product by ZTE. The migration towards more flexible terminals that should use as common components as possible has increased the trend towards considering SDRs as the future technology also for commercial equipment. In fact, in the base station domain this paradigm shift is already evident.

In 2008, ETSI (European Telecommunications Standards Institute) started a Technical Committee to conduct feasibility studies for the standardization of a wider concept of SDR technology called RRS (Reconfigurable Radio Systems), which are defined as follows: "*The group of technologies for Cognitive Radio and for Software Defined Radio are all technologies for Reconfigurable Radio Systems (RRS). Such systems exploit the capabilities of reconfigurable radio and networks and self-adaptation to a dynamically changing environment, with the aim to ensure end-to-end connectivity.*"

ETSI TC RRS is composed by four working groups: WG1 System Aspects, WG2 terminal architecture, WG3 Functional architecture and WG4 Public Safety. ETSI TC RRS has already produced a number of technical reports in all four working groups.

WG2 has presented a new SDR reference architecture for mobile devices (i.e. handheld) in ETSI TR 102 680 [95], where an Unified Radio Application (URA) interface is presented. The SDR functional architecture is presented in Figure 1-6:

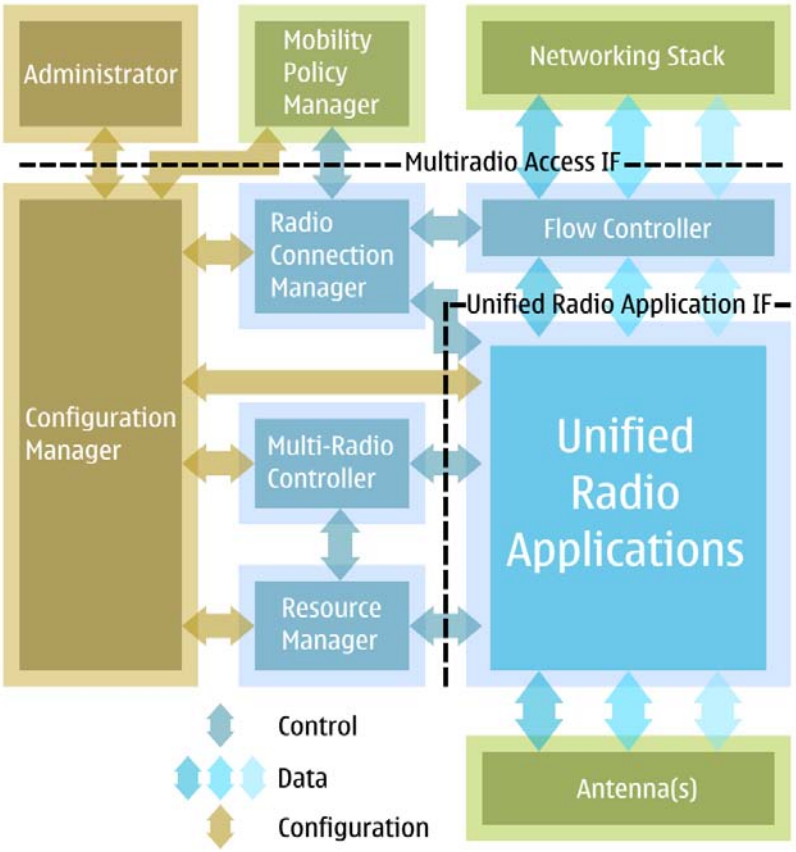


Figure 1-6 Functional architecture of SDR Equipment (from ETSI TR 102 680)

This architecture is composed by the following interfaces and subsystems:

- *Multiradio Access Interface*. Mobile device applications can access all radio computer services provided at the Multiradio Access Interface. The services include normal connectivity to Peer Equipments with uni/bidirectional data transfer. Services provided by more specific radio applications, such as positioning and broadcasting are also provided in the same uniform manner.

- *Unified Radio Application Interface*. The purpose of the Unified Radio Application Interface (URAI) is to harmonize the behaviour of radio applications towards the radio computer operating system. To achieve this, all radio applications will access and provide a well-defined set of services specified in the Unified Radio Application Interface.
- *Administrator*. The administrator is responsible for downloading and activating software waveforms and modules on the SDR platform. The administrator is also responsible for storing and activated software profiles (e.g., specific set of parameters of the waveforms).
- *Mobility Policy Manager.* It maintains the user policies and preferences on different radio access types.

The Multiradio Access Interface has been further described in ETSI TR 102 839[96], where the main workflow have been identified and described.

Cognitive Radio can support Flexible Spectrum Management approaches, which are radically different from the current static spectrum allocation of bands to specific wireless services. Availability of spectrum is not a critical issue in the military domain, but it is becoming increasingly important in the commercial domain.

Various regulatory and standardization bodies have investigated cognitive radio and dynamic spectrum allocation in the commercial domain. An early cognitive radio application is the so called "*White spaces*" which refers to portions of the RF spectrum, which are not used by broadcasters. To improve spectrum utilization, it is possible to transmit and receive in "white spaces" with the clause that there should not be harmful interference to broadcasters and other primary services.

In USA, in 2008, FCC voted to approve the unlicensed use of white spaces. The Second Report and Order [6] described the position of the FCC and the concept of the FCC-mandated database to determine which channels are available for use at a given location. The report also described the spectrum sensing function, which "white spaces" devices must support to monitor the spectrum and detect the presence of broadcasters. In September 2010, the FCC released a *Memorandum Opinion and Order* that determined the final rules for the use of white space for unlicensed wireless devices. The new rules removed mandatory sensing requirements, which greatly facilitates the use of the spectrum with geolocation based channel allocation.

In Europe, the ECC WGSE (Spectrum Engineering) has set up the special *project SE43* dealing with cognitive radio matters in May 2009. The objective of SE43 is to complete the ECC Report "Technical and Operational Requirements for the Possible Operation of Cognitive Radio Systems in the 'White Spaces' of the Frequency Band 470-790 MHz". The main focus of the report is on coexistence with incumbent or primary systems. The report defines different scenarios for CR operation in terms of WSD types (personal/portable, home/office and public access points) and also discusses the three well known types of cognitive techniques: spectrum sensing, geo-location and beacons. The report is focussed on protection of four possible incumbent systems: broadcast systems (BS), Program making and special events (PMSE), radio astronomy (RAS) and aeronautical radio navigation systems (ARNS). Comprehensive data on possible sensing and separation distances are given, and ends in operational and technical characteristics for white spaces devices to operate in the band. An estimate of available white space is also included.

In February 2010 the European Commission DG INFSO Radio Spectrum Policy Group (RSPG) published a first report on cognitive technologies. The report had the objective of informing policy makers in Europe as early as possible of the discussions and challenges raised by Cognitive technologies. The report provided an overview of cognitive radio technologies, described new spectrum management approaches based on cognitive radio and identified challenging issues that require further attention. The use of cognitive radio technologies is seen as an enabler providing more efficient spectrum sharing and providing more dynamic access to spectrum.

In October 2010, RSPG published a new document "RSPG Opinion On Cognitive Technologies" RSPG10-348 to identify benefits and challenges of implementing CR at Community level. In particular, the opinion investigates the need for coordination at EU level on cognitive radio technologies.

In particular, the scope of this opinion encompasses:

- high-level approach at each cognitive feature studied within a common European framework;
- considerations on the possible merits of taking further regulatory steps;
- near-term regulatory action that needs to be taken to enable any cognitive technology;
- need to establish a harmonised basis and if so the actions necessary.

In the UK spectrum regulators OFCOM has also permitted in [78] cognitive access to TV spectrum bands as long as they would not cause harmful interference to licensed users.

2. Existing SDR and CR platforms

During the last decade, the development of software-defined radios (SDR) has seen a remarkable thrust. While a SDR is intended to be controlled and reconfigured using software that can be easily downloaded, setup and upgraded, it remains closely dependent from a RF front-end allowing it to transmit and receive voice, data and video. Meanwhile, RF front-ends are technology-dependent. Their performance in terms of linearity, power consumption and bit error rate, changes when the carrier frequency changes too. And SDRs intended to be multistandard radios, are particularly hindered by these properties. In fact, most of today's developments in SDR design have focused on the baseband section. But, little emphasis was placed on the radio-frequency (RF) part of radios. Despite the fact that innovative RF techniques of spectrum management, sensing and usage were investigated and new RF building block designs for SDR applications were presented, little in-depth work has been initiated in order to overcome the challenging issues and bottlenecks on the RF side.

The main objective here is to describe the existing SDR and CR platforms and testbeds in the context of ACROPOLIS project. The goal of this activity is to produce different experimental real-time validation platforms as proof-of-concept for signal- and packet-level cooperation and end-to-end information transport in the form of multicasting. Those platforms will be developed on eight platforms, which have been deployed by ACROPOLIS partners. We will describe also in Sections2.8 and 2.9 some other platforms and testbeds.

2.1 GNU radio software toolkit

GNU Radio [80] is an open source software toolkit for developing software defined radio applications. It enables definition of transmitting waveforms and demodulation of received waveforms through software processing, as opposed to analogue circuitry processing or analogue circuitry combined with digital chips processing. The basic principle functions in a two tired architecture with creation of signal processing blocks in the first tire and their interconnection in the second tire. More generally, GNU Radio encompasses the following entities:

- An API for creating signal blocks (C++/Python)
- A runtime environment for signal processing
- A library of signal processing blocks
- User scripts and applications
- Hardware drivers (Universal Software Radio Peripheral (USRP)/USRP2-VRT (VITA Radio Transport protocol))
- An application for creating flow graphs (GRC)

Figure 2-1 depicts the position of the GNU Radio software in a practical implementation. Its intention as a software defined radio tool is to be placed as near as it can be to the antenna.

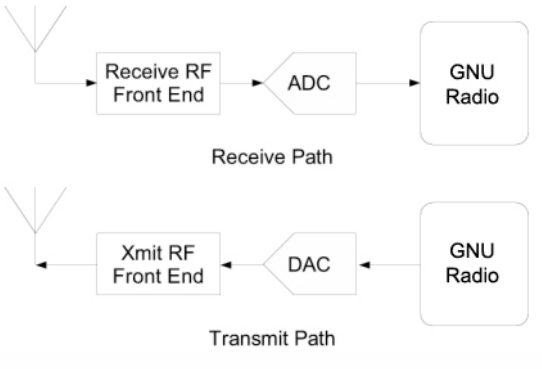


Figure 2-1 GNU Radio software development tool implementation [37]

GNU Radio's programming environment is organized around the principle of constructing a signal processing graph that describes the data flow in the software radio. Signal data flows are constructed using interconnected signal processing blocks and are primarily written using the Python programming language. Signal processing blocks are functional entities implemented in C++ which operate on infinite data streams flowing from a number of input ports to a number of output ports specified per block. GNU Radio provides a large software library of individual signal processing routines as well as complete signal processing blocks, including various modulations (GMSK, PSK, QAM, OFDM), error-correcting codes (Reed-Solomon, Viterbi, Turbo Codes), signal processing constructs (optimized filters, Fast Fourier Transport (FFT), equalizers, timing recovery) and scheduling. Figure 2-2 depicts an example of a signal processing blocks interconnection using USRP2 hardware for spectrum sensing [26]. It comprises six signal processing blocks, designated with the "gr" prefix, and one custom made USRP interfacing block with the "usrp2" prefix. The respective functionalities of each block are:

- usrp2_source_32fc - creates the USRP2 source and controls the hardware (sets up sampling rate and tuning frequencies);
- gr_stream_to_vector - converts a stream of complex samples to vector of complex samples;
- gr_fft_vcc - calculates a FFT transform on input complex samples;
- gr_complex_to_mag - calculates magnitude (amplitude) on complex samples;
- gr_fft_vfc - calculates a FFT transform on input real samples;
- gr_complex_to_mag_squared - calculates squared magnitude (power) on complex samples;
- gr_energy_detector_f (custom made) - selects between different detector types and initiates switching between frequencies.

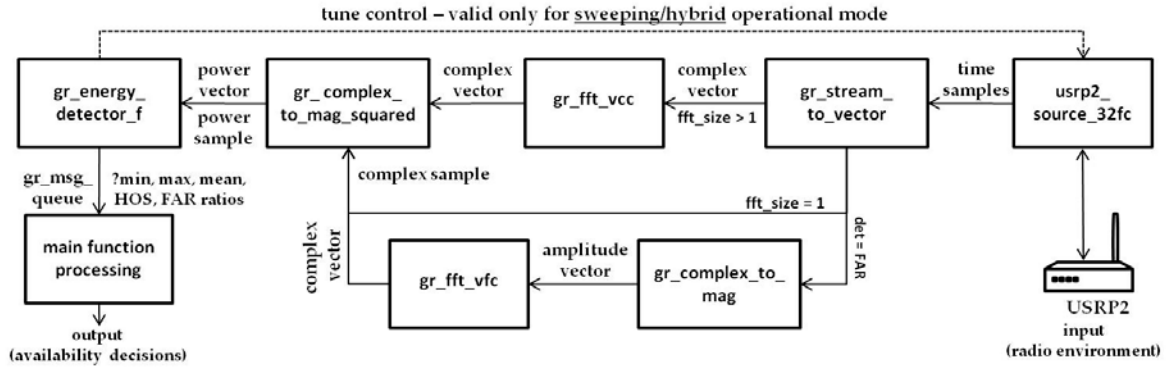


Figure 2-2 An example of signal processing blocks interconnection [33]

While GNU Radio is hardware-independent, it directly supports hardware front-ends via specific source and sink signal blocks. Due to its high integration in GNU radio and low cost, the USRP product family is usually coupled with this software framework. Figure 2-3 illustrates the interconnection of the USRP hardware with the GNU Radio. Simply stated, the GNU Radio software sets the transmission parameters on the USRP motherboard using the adequate interface (USB 2.0 or Ethernet). Usually, when developing a cognitive radio testbed, these parameters are not determined by the user, but stem from the entire spectrum management process.

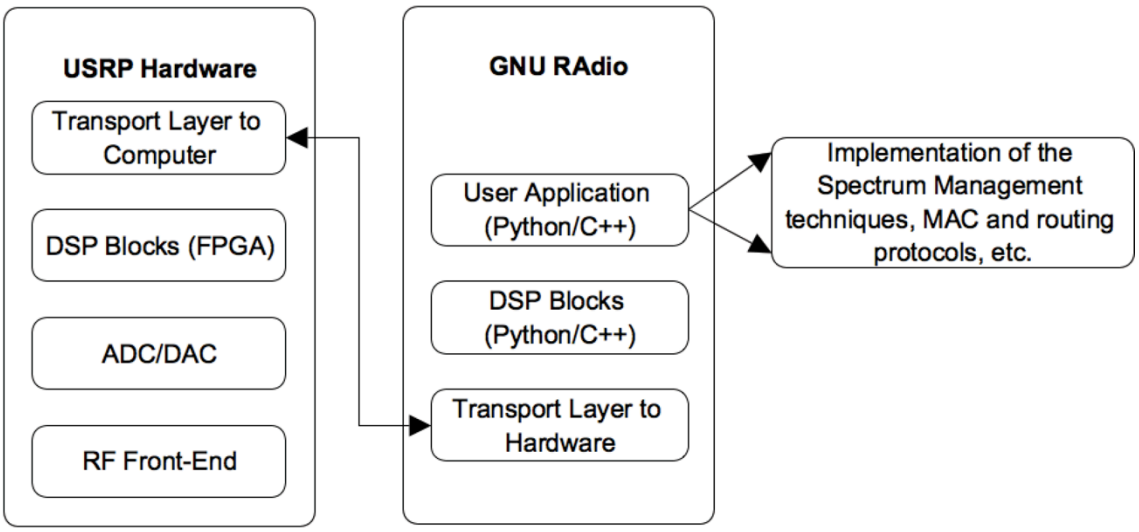


Figure 2-3 Block diagram of the USRP-GNU Radio framework [37]

Signal graphs construction can be simplified using the GNU Radio Companion (GRC). The GNU Radio Companion is a graphical user interface which represents an alternative to the Python-based connection of signal processing blocks. It uses Cheetah templates to generate the Python source code for the flow graph. GNU Radio Companion can generate source code for GUI and non-GUI flow graphs as well as hierarchical blocks. The 3.2 release includes an efficient way for definition of variables and blocks, as well as internal structure of the saved flow graph files.

Every block in GRC has a corresponding XML file that contains parameters, IO ports, and a template for code generation. The id key and file name of each Extensible Markup Language (XML) file matches up exactly with the name of the GNU Radio block to ensure future

portability. GRC validates all blocks definitions upon execution and will exit with error if any definitions fail the validation. Variable blocks show up in the flow graph and act like any other block, except that they have no IO ports. The variable block maps a unique id (variable name) to a particular value. GRC also has several graphical variable blocks that allow one to create WX GUI flow graphs with graphical controls using sliders, text boxes, buttons, drop downs and radio buttons. GRC creates an XML file from the created flow graph that is translated to Python code. References to the signal processing blocks of the GNU Radio library are also included in GRC by the means of XML definition files. This creates the possibility to include custom made signal processing blocks by defining an XML file for the new blocks. Figure 2-4 shows a screenshot of GRC with three signal processing blocks: two signal sources and one audio sink [38]. As mentioned before, signal processing blocks connected together represent a flow graph.

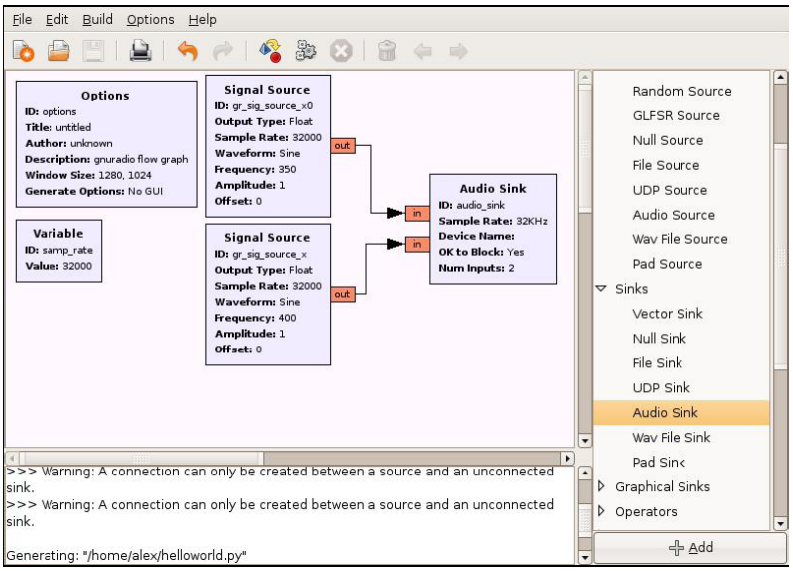


Figure 2-4 GNU Radio Companion screenshot [38]

2.2 IRIS platform

IRIS [32] is a platform that has been developed since 2004 in CTVR (mostly by their Trinity College site), Dublin. Especially after redesign in 2008 it provides functionalities crucial for efficient implementation of cognitive radio network:

- Portability between different OSs and CPUs
- Fast and reliable radio reconfiguration on run-time, many reconfiguration mechanisms
- Support for PHY and higher layers of network stack
- Parallel computing capability
- Possible utilization of high speed external computing units e.g. FPGA, Cell Broadband Engine (CellBE)

The radio model is components-based so, similarly to GNU radio, the transmission chain is build of separate components (written usually in C++ language and precompiled) connected

using XML configuration file (in GNU radio Python takes similar role) on run-time. The physical frontend used natively by this platform is very common USRP product family. All components have to work according to one of provided “engines” so that communication between components is possible. As the requirements on reconfigurability and data flow characteristic is component dependent there are three engines prepared (although other can be designed later on):

- **Scheduled PHY Engine**

  It is designed for high data rate operation, usually close to the analogue front-end, so the Synchronous Dataflow Model of Computation was used. To provide high performance some limitations had to be introduced: fixed data rate on the input and output of the component, unidirectional data flow. Although reconfigurability is still possible it requires computationally complex recalculation of the architecture so it is not recommended. An example of such component can be digital filter or interpolator. Writing to output and reading from input uses buffers.

- **Flexible PHY Engine**

  As the name indicates, this layer is also designed for PHY layer, but the reconfiguration is less problematic. An example can be OFDM modulator component. The dataflow is, as previously, unidirectional and behaves according to Dataflow Process Networks Model of Computation. Each component is executed asynchronously. It is also not required to keep input and output data rates static as it would be hard to keep e.g. when the OFDM modulator changes modulation order. As previously communication with other components is based on input and output buffers.

- **Network Stack Engine**

  This engine is to be used by MAC layer or network layer components. The data flow can be bidirectional inside each component. Moreover, as here stream is divided into frames or packets and e.g. only headers are added (so there is no need to process every bit separately) the service of inputs and outputs is based on pointers, not data buffers used before. However, the Model of Computation is still Dataflow Process Network.

The use of many different engines might not only be required to fit components characteristic but also to increase speed. Each engine (It is worth mentioning that many instances of the same engine type can be used in the same radio) is run as separate thread, therefore parallel computing on multicore CPUs may improve performance. If this gain is not sufficient FPGA or CellBE platforms can be used by using appropriate “software wrapper” component inside radio chain that takes care of interfacing with those external platforms.

The main advantage of the IRIS platform is the possibility of the run-time reconfiguration, i.e. the ability to modify various functional parameters of the platform during operation (execution of the implemented program).These changes can be applied to the parameters of certain components which are defined in the corresponding xml file (such as the length of cyclic prefix of the OFDM symbol). Moreover, the each component in the processing chain can be set as active or inactive. The reconfiguration can be done mainly twofold: by means of the external or internal trigger. In the first case the user changes the xml configuration file. Than the dedicated IRIS block (called *Reconfiguration manager*) compares the actual

configuration file with the new one and updates the parameters (possibly without stopping the running transmission). In the latter case, which is the more sophisticated one, the dedicated controllers check if the values of the certain parameters are within the assumed range (e.g. The BER controller verifies if the actual BER is below the target level). Based on this the controllers can modify the system configuration. In general, these controllers are blocks working in parallel to the main radio chain that can observe behaviour of the whole radio and react on it. An example might be the abovementioned BER controller that observes BER on the output of the demodulator and increase or decreases the modulation order.

Although, above presented description is unquestionably positive there are also some drawbacks. As this platform, to the best of authors' knowledge, is not freely available the number of available components and the size of users' community is limited. There is also no exhaustive documentation that might cause a problem while dealing with more refined platform features.

2.2.1 Short overview of the IRIS demonstrators

Since IRIS is a software architecture for building reconfigurable radio networks, a number of interesting application of it can be found in the literature, e.g.[33], [34], [35] Moreover, many demonstration systems based on the IRIS have been built and presented at various conferences, just to mention a few of them [32]:

- Demonstrator presented at DySPAN conference in 2007 and entitled: *Cyclostationary signatures for detection, identification, and carrier frequency estimation of OFDM-based waveforms*
- Demonstrator presented at DySPAN conference in 2008 and entitled: *A reconfigurable FPGA-based system using a single-carrier waveform, employing sensing for carrier-frequency estimation*
- Demonstrator presented at SIGCOMM conference in 2009 and entitled: *A reconfigurable FPGA-based system employing sensing for carrier-frequency estimation and adaptive coding*
- Demonstrator presented at DySPAN conference in 2010 and entitled: *A DSA network using reconfigurable pulse-shaped OFDM waveforms to control out-of-band (OOB) emissions*

2.2.2 Exemplary experiment realized using IRIS software architecture

As it has been already mentioned, IRIS architecture is software-oriented, thus various implementation using the IRIS modules can be realized and presented. The main idea of the demonstrator prepared at OUT was to evaluate (from the perspective of the cognitive radio and the amount of introduced interference to the primary user) the effectiveness of the proposed spectrum shaping methods applied to the non-continuous OFDM (NC-OFDM) signal. The CU's signal is transmitted in the interleaved spectrum, i.e. the frequency band utilized by the cognitive user is fragmented due to the presence of a couple of narrow-band primary transmitters (e.g. the wireless microphones or other programme making and special events (PMSE) devices). Since the width of the frequency band used by the certain primary user is narrow, the cognitive terminal has to fulfil strong requirements with regard to the interference introduced to the primary user band. Thus, efficient out-of-band reduction

techniques have to be applied in order to keep the interference power under the allowable level. Some spectrum shaping algorithms have been developed at PUT and implemented on the IRIS platform.
In Figure 2-5 the demonstration system based on the IRIS architecture has been presented. The cognitive transmitter that uses the proposed spectrum shaping algorithms has been implemented in the IRIS platform, i.e. the multicarrier signal based on the NC-OFDM technique has been generated on the computer, send to the USRP board and transmitted. The power spectral density plot of the received signal has been created on the spectrum analyzer, thus the power of the out-of-band emission can be measured. The screenshot presenting the settings of the working IRIS platform has been presented in Figure 2-6, while the exemplary results that show the effectiveness of the implemented spectrum shaping algorithm in the presence of one PU PMSE device have been shown in Figure 2-7. In the latter the black and blue lines correspond to the PSD plot of the NC-OFDM signal without and with spectrum shaping algorithms, respectively.

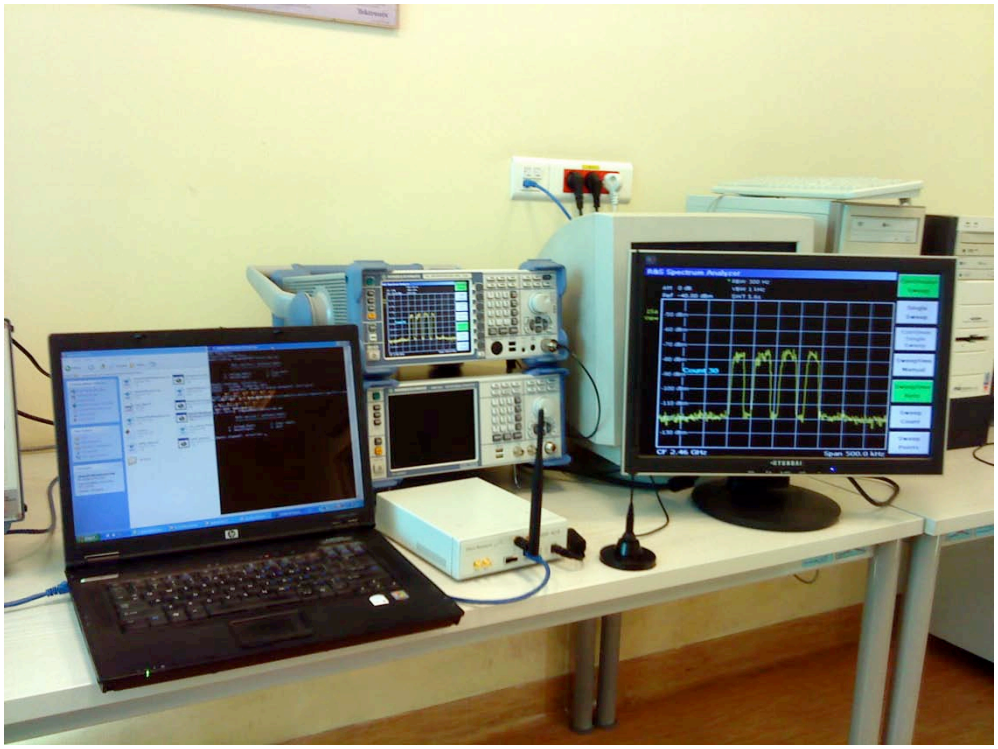


Figure 2-5 Demonstration system based on the IRIS architecture developed at PUT

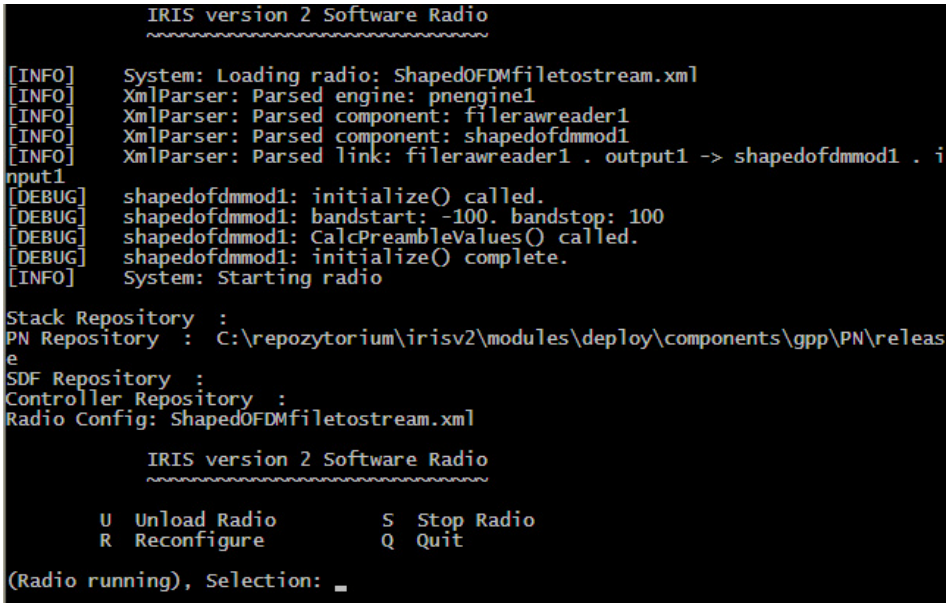


Figure 2-6. Screenshot of the running IRIS platform

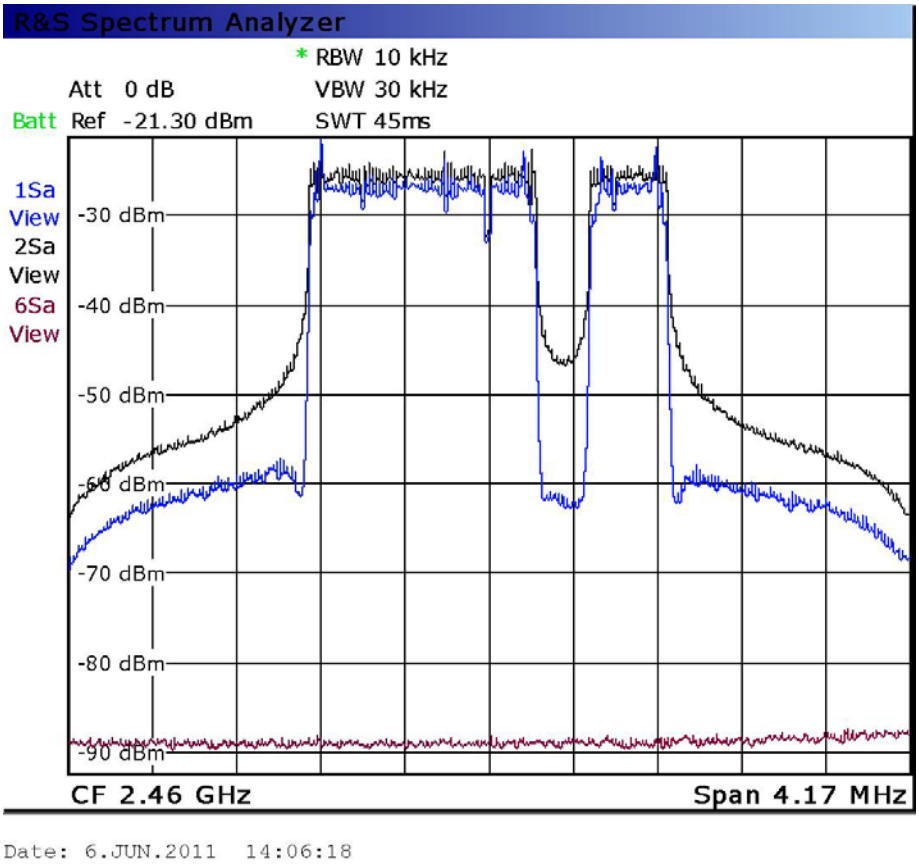


Figure 2-7 Screenshot from the R&S spectrum analyzer

As it has been already mentioned, the IRIS platform bases on the XML, i.e. the structure of the simulator is defined by means of XML file. The very simple exemplary XML file prepared for the auto-test of the demonstration platform presented in this subchapter is shown in Figure 2-8. In that case the data to be sent by the cognitive user have been intentionally stored in a file. Also the received data have to be stored on the computer. One can observe that the new project called “Radio1” has been created that consists of four components: “filereader”, “simpleofdmmod1”, “simpleofdmdemod1” and “filewriter1”. The parameters

of these blocks have been also defined, i.e. filename, size of the blocks and data type used. The relations between the certain blocks have been also defined at the end of the XML file. Let us stress that in that simple and intuitive way one can define even very complicated systems on the IRIS platform.

Figure 2-8. Exemplary XML file used for definition of the cognitive transmitter implemented on the IRIS platform

```
<?xml version="1.0" encoding="utf-8" ?>
<softwareradio name="Radio1">
  <engine name="pnengine1" class="pnengine">
    <component name="filerawreader1" class="filerawreader">
      <parameter name="filename" value="testdata.bin" />
      <parameter name="blocksize" value="1024" />
      <parameter name="datatype" value="uint8_t" />
      <port name="output1" class="output" />
    </component>
    <component name="simpleofdmmod1" class="simpleofdmmod">
      <parameter name="debug" value="false" />
      <parameter name="bitsperdatasymbol" value="4" />
      <port name="input1" class="input" />
      <port name="output1" class="output" />
    </component>
    <component name="simpleofdmdemod1" class="simpleofdmdemod">
      <parameter name="debug" value="false" />
      <port name="input1" class="input" />
      <port name="output1" class="output" />
    </component>
    <component name="filerawwriter1" class="filerawwriter">
      <parameter name="filename" value="out.txt" />
      <port name="input1" class="input" />
    </component>
  </engine>
  <link source="filerawreader1.output1" sink="simpleofdmmod1.input1" />
  <link source="simpleofdmmod1.output1" sink="simpleofdmdemod1.input1" />
  <link source="simpleofdmdemod1.output1" sink="filerawwriter1.input1" />
</softwareradio>
```

2.3 Universal Software Radio Peripheral (USRP)

One of the most popular Software Defied Radio platform for testing of Cognitive Radio implementations is the Universal Software Radio Peripheral (USRP), developed within the Ettus Research™ LLC founded by Matt Ettus in 2004 [26]. Although from the February 2010 the firm operates as a wholly owned subsidiary of National Instruments Corporation, the original name, USRP, has been left unchanged. Due to its various advantages, USRP has been classified as the most popular, cognitive radio demonstration platform devoted for generic purposes, i.e. which is not dedicated to the specific implementation [27].

The USRP platform provides to the wide spectrum of engineers and system designers fast and efficient way for development and testing of the particular cognitive procedures or even whole flexible radio systems. This is possible due to the simplicity of selection of the certain frequency band assumed for transmission for a considered cognitive system. Such desired flexibility in the wireless system design has been ensured by careful and proper preparation of the USRP architecture that combines high implementation efficiency with high simplicity of usage. The generic diagram of the USRP is illustrated in Figure 2-9. The detailed model of the USRP has been proposed in [28]; in this paper the comprehensive

emulator of the USRP1 and RFX2400 daughterboard has been described as well as the main advantages and weakness of this platform have been pointed out.

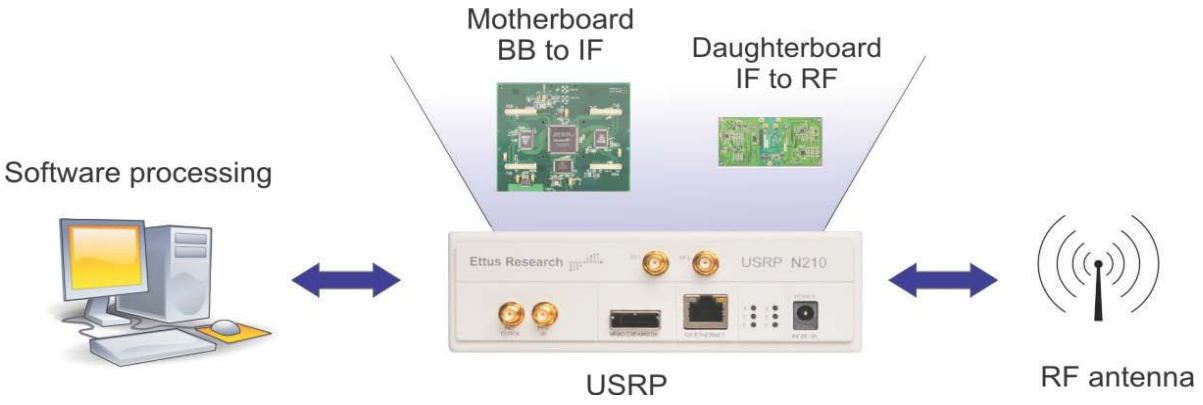


Figure 2-9 Generic diagram of the USRP-based wireless communication transceiver

When analysing the above figure, one can observe that the whole software processing is done directly on the computer and then sampled and send (via USB or Ethernet) to the so-called motherboard, where the input signal from the computer is modulated to the intermediate frequency (IF). Afterwards, the IF signal is converted from digital to analogue form and then shifted directly to the certain radio frequency (RF) band (assumed for real transmission) on the connected daughter board and send to the antenna. A variety of daughter boards have been developed and offered to the market within the Ettus Research™ LLC company, each devoted to the particular frequency band. All of the daughter boards are shortly described in the Table 2-1.

Table 2-1 The overview of the USRP daughter-boards offered on the market

Name	Covered frequency band	Additional information
BasicTX BasicRX	1 – 250 MHz IF	These boards are designed for use with external RF front-ends as anIF interface The ADC inputs and DAC outputs signals are transformed and coupled directly to SMA connectors with no mixers, filters, or amplifiers
LFTX / LFRX	DC to 30 MHz	Similar to BasicTX/RX, however differential amplifier (instead of transformers) and 30 MHz low pass filters for antialiasing have been applied
TVRX2	Dual 50 MHz to 860 MHz	This board offers two independent low-IF receivers covering the VHF and UHF bands with 10 MHz baseband bandwidth.
DBSRX2	800MHz – 2.4GHz (ISM band is not included)	This board is a complete receiver system features a software controllable channel filter that can be made as narrow as 1 MHz, or as wide as 60 MHz. MIMO communication is also supported - an active antenna can be powered via the coax.
RFX900	750 MHz – 1.05 GHz	Full transceiver of the typical transmit power: 200 mW
RFX1200	1150 to 1450 MHz	Full transceiver of the typical transmit power: 200 mW
RFX1800	1.5 GHz to 2.1 GHz	Full transceiver of the typical transmit power: 100 mW
RFX2400	2.3 GHz- 2.5 GHz	Full transceiver of the typical transmit power: 50 mW
WBX	50 MHz to 2.2 GHz	Full transceiver of the typical transmit power from 30 mW to 100 mW; Wide frequency range is covered, thus this board is suitable for testing of various scenarios including white spaces, broadcast television, public safety, land-mobile communications, wireless sensor networks, cell phones etc.
SBX	400 MHz to 4.4 GHz	Full transceiver of the typical transmit power from 30 mW to 100 mW; Wide frequency range is covered, thus this board is suitable for testing of various scenarios including cellular, WiFi, WiMax, microwave S band and the 2.4 GHz ISM applications
XCVR2450	2.4 to 2.5 GHz and 4.9 to 5.9 GHz	Full transceiver of the typical transmit power: 100 mW The XCVR2450 covers the 2.4GHz ISM band and the entire 4.9 to 5.9 GHz band, thus including the public safety, UNII, ISM bands

The main features of all daughter-boards highlighted by the producer can be summarized in a nutshell as follows:

- all of the boards support MIMO communications
- the I/Q architecture has been assumed
- the full-duplex transmission has been allowed (clearly except the XCVR2450 board in case of transmitters)
- the transmit/receive switching is supported
- theoretically the transmit bandwidth can be as high as 30MHz
- all functions of the daughter-boards are controlled from the software or from FPGA.

From the implementation point of view, the USRP is featured with the simply downloaded open-source software called GNU Radio ([29], see also Section 2.3). Such a software package covers not only the complete radio-, but also the signal processing area. Once installed is ready to used, however, one has to remember that the software (including firmware) have been optimized for Linux-based operation systems. It is worth mentioning that USRP has been already utilized in the solid Open Source projects, like OpenBTS [30] and OSSIE [31]. Beside the OpenSource applications, a couple of commercial solutions have been proposed:

- LabVIEW:
  the National Instruments informs that the drivers for the entire USRP family of products are currently in development;
- MATLAB/SIMULINK®:
  USRP is already supported twofold, first the drivers for the USRP v.1 already exists, second the Simulink Communications Blockset (release R2010b) supports the USRP2.

Let us also stress that, as the authors said, the “true value of the USRP is in what it enables engineers and designers to create on a low budget and with a minimum of effort” and “the combination of flexible hardware, open-source software and a community of experienced users makes it the ideal platform for your software radio development”.

2.3.1 USRP 1

In this first version of USRP, high sample-rate processing is done in the FPGA, while lower sample-rate processing is performed in the host computer. The board is connected with the user-computer via the USB2.0, thus allowing for transmitting of 480 Mbps. The input signal (from the computer side) is interpolated in the digital way (digital upconverters have been used) to 128Msps. Contrarily, the received signal from the USRP is digitally downconverted (decimated) from the 64Msps to the required sampling rate. In the transmitter chain the 14-bit Digital-to-Analogue Converter has been used, while in the receiver part the 12-bit Analogue-to-Digital Converter has been implemented.

2.3.2 USRP 2

USRP2 [22][33][39] is the successor of the first USRP hardware platform updated with more powerful hardware and features compared to the previous version. The main characteristics of the second version of the USRP motherboard are:

- *Gbit Ethernet interface* for connection to the host computer - unlike USRP1, where the USB interface limits the maximum RF bandwidth to 8 MHz, the GigE of the USRP2 motherboard supports maximum transfer rate of 125 MB/s which is equivalent to about 30 MS/s (when the IQ samples are sampled with 2 bytes each).
- *ADCs and DACs (a pair of each – IQ sampling) with sampling rate of 100 MS/s and 400 MS/s and quantization with 14 and 16 bits*, respectively. These capabilities allow higher RF bandwidth and increased dynamic range and precision compared to the USRP1.
- *Digital downconverters and upconverters with programmable rates*. The decimation and interpolation factors are in the range 4-512 resulting in RF bandwidths of ~195 kHz up to 25 MHz.
- *Spartan 3 FPGA* with *50 MHz 32-bit RISC CPU* performing the high-rate operations of digital up and down conversion. The lower sample rate operations are performed on the host PC side. The USRP2 motherboard has *1 MB high-speed on-board SRAM* memory, while the FPGA and firmware codes are read from a *SD card*.
- Unlike the previous version, a single USRP2 cannot perform as a MIMO transmitter/receiver. However, multiple USRP2 systems can be connected togetherto form fully coherent multiple antenna systems for*MIMO with as many as 8 antennas*.
- The modular architecture of the USRP2 supports a wide variety of RF daughterboards, same as in the first version of the hardware.

2.3.2.1 USRP2 Testbed at UKIM

Figure 2-6 presents the testbed at the Faculty of Electrical Engineering and Information Technologies, Ss. Cyril and Methodius University in Skopje. It comprises six USRP2 motherboards which can be combined with four RFX2400 daughterboards, two WBX daughterboards and two RFX400 daughterboards, providing the capabilities to sense the 2.4 GHz ISM band, the bands up to 2.2 GHz and the 2.4 GHz ISM band, respectively. The USRP2 testbed is used mainly for cognitive radio research activities at UKIM. It provides the options to implement and test various sensing algorithms, cooperative/collaborative sensing and sharing algorithms, as well as to convey spectrum measurements in various bands. As an instance the Figures 2-7 a) and b) present snapshots of the real-time duty cycle and the real-time power waterfall captured with a USRP2 device at UKIM's premises in the 2.4 GHz ISM bands.

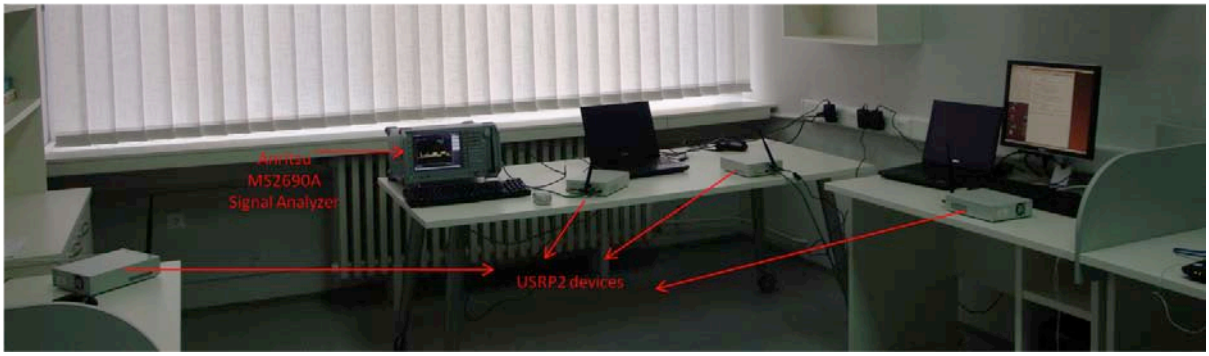
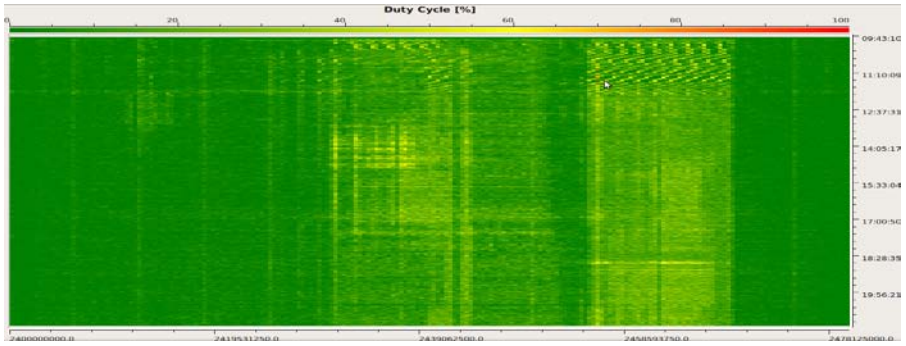
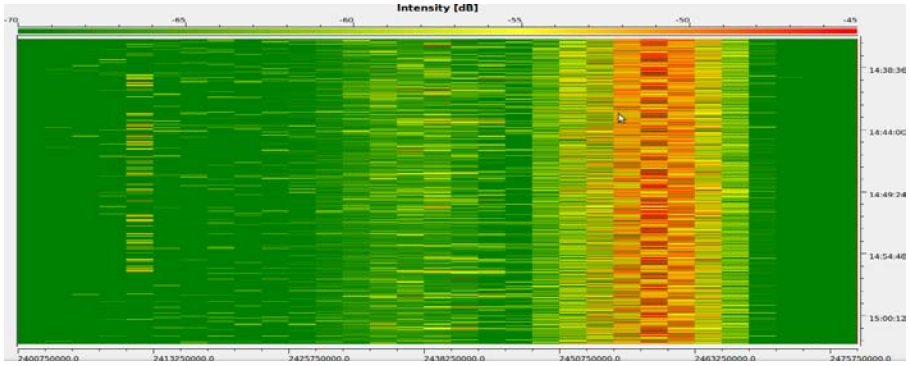


Figure 2-10 UKIM's USRP2 Testbed



(a)



(b)

Figure 2-11 Waterfalls of the activity in the 2.4 GHz ISM band captured with a USRP2 device between 9:30 and 21:30 hours during a working day at UKIM's premises in terms of a) Duty cycle and b) Received signal power

2.3.3 USRP n210

The USRP N210 has been introduced in November 2010. It follows in general USRP 2 architecture assumptions, but some improvements were introduced. It uses the same DACs, ADCs and host interface (listed in table below), so the spanned bandwidth is the same. As each real sample on the bus consists of 16 bits, the whole I-Q symbol is 32 bits. Taking into account maximum of 25MHz of spanned bandwidth, the usable throughput over Gigabit Ethernet cable is 25*32Mb/s=800Mb/s. It is worth mentioning that as digital-analogue converter has 4-times embedded interpolation at its input the maximum stream of 100MS/s is possible. The program defined interpolation (from 4 to 512) is from user point of view adjusted to keep this value constant.

The advancements of this board over USRP2 are as follows:

- The FPGA image is reconfigurable over Ethernet and stored in build-in flash card. No SD card used and complicated burning required.
- More powerful FPGA: Xilinx Spartan 3A-DSP3400A. According to December 2010 FPGA code there are 63% of general logic, 66% of memory and 88%of DSP resources free for future development.
- Improved synchronization support: external clock can be connected or (optionally) GPS synchronization module can be used.

Recently, the USRP N200 was introduced that provides similar functionality, but is cheaper as less expensive Xilinx Spartan 3A-DSP1800 FPGA was used.

Table 2-2 The summary of main features of all considered USRP boards

Feature	**USRP 1**	**USRP 2**	**USRP N210**
Interface	USB 2.0	Gigabit Ethernet	Gigabit Ethernet
Bandwidth available while complex samples used	8MHz	25MHz	25MHz
DAC	128 MS/s, 14bit	400MS/s 16-bit	400MS/s 16-bit
ADC	64 MS/s 12 bit	100MS/s 14-bit	100MS/s 14-bit
Firmware update	Through USB (during each start-up)	SD card	Through Ethernet
MIMO Capable	Yes (not fully)	Yes	Yes

2.3.3.1 USRP1/2/N210 Testbed at RWTH Aachen University

The USRP devices are one of the most used platforms within ACROPOLIS consortium, and it seems to be also generally the most popular one in Europe in cognitive radio context. There exists several testbed, and out of ACROPOLIS partners RWTH Aachen University has a widest deployment of USRP1, USRP2 and USRP N210 devices having overall over 50 devices used for relevant research and teaching purposes. The teaching is mostly focused on using it as a basic SDR platform for digital communications and packet radio teaching in laboratory environments. In the research, the platforms have been used for spectrum sensing, point-to-point link design (e.g. OFDM transceiver design testing), and to deploy small- and mid-scale test networks (5-30 nodes deployments).

2.4 Wireless Open-Access Research Platform (WARP)

Wireless open-Access Research Platform (WARP) [42] is a custom platform for research in advanced wireless algorithms and applications. The platform consists of both custom hardware and FPGA implementations of key communications blocks. The hardware consists of FPGA-based processing boards coupled to wideband radios and other I/O interfaces; the algorithm implementations already include a flexible OFDM physical layer [48]. A couple of simple MAC protocol implementations, realized in software, are also provided. Both the hardware specifications and algorithm implementations are freely available to the research community.

2.4.1 Hardware Architecture

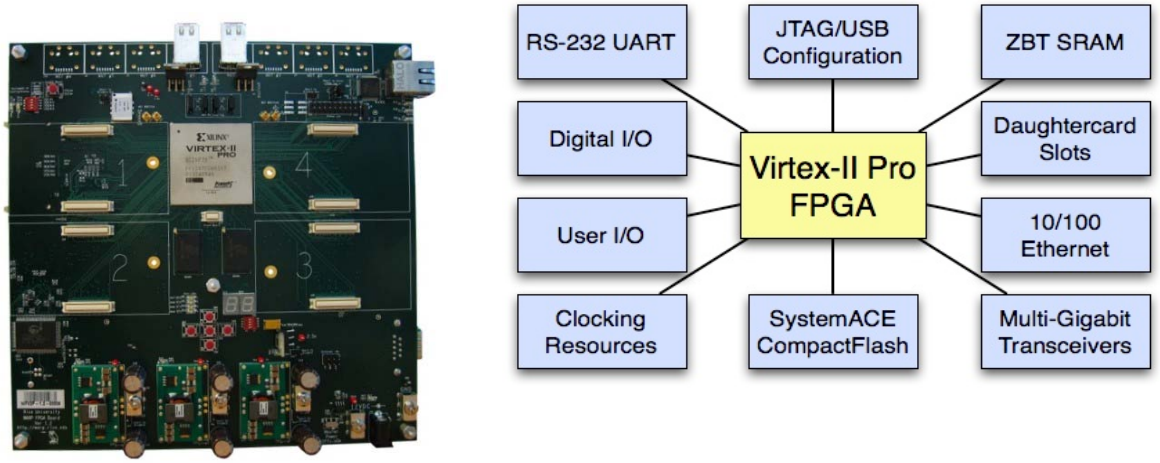


Figure 2-12 WARP Hardware [42]

The WARP v1.2 FPGA board is a 8"x8" PCB built around a Xilinx XC2VP70 Virtex-II Pro FPGA. The WARP FPGA board operates from a single external 12v supply. The FPGA can be configured through an on-board USB configuration circuit by connecting the WARP FPGA Board's USB port directly to the PC using a standard USB cable. One can also configure the FPGA using an external Xiilnx JTAG cable. The WARP FPGA board has a standard JTAG connector which is compatible with both the Xilinx Parallel IV and Platform USB cables. In addition, the WARP FPGA board also includes Xilinx's SystemACE CompactFlash chip for managing the configuration process of the FPGA. The SystemACE chip acts as an interface between the FPGA and a standard CompactFlash slot. The SystemACE automatically configures the FPGA on powerup or when the manual reset button is pushed. Multiple configuration files can be stored on the CF card and selected according to a dipswitch located near the CompactFlash slot.

The WARP FPGA Board has four daughtercardslots which ultimately supports 4x4 MIMO communication. The four slots are electrically and mechanically identical. The WARP hardware supports any combination of daughtercards in the four slots. However, a given FPGA design will require a specific arrangement of daughtercards once synthesized. On-the-fly reconfiguration of FPGA is not enabled in the current implementations.

2.4.2 Software Frameworks

There are two major open-sourced software reference designs provided by Rice University. These reference designs are commonly used for both research on PHY/MAC designing for software defined radio and educational purposes. Xilinx's Embedded Development Kit is needed to build these designs locally.

2.4.2.1 OFDM Reference Design

It is a full project with MIMO-OFDM, ALOHA/CSMA MAC and Ethernet hub functionality. The WARP OFDM Reference Design implements a real-time network stack on a WARP node. The design includes a MIMO OFDM physical layer and flexible MAC interface for building custom protocols. This design demonstrates the full MAC/PHY capabilities of WARP. All processing (hardware control, signal processing, MAC protocol) is executed in real-time by each WARP

node. Serial communication is included in the design. PCs can be attached to WARP nodes via Ethernet, but are used only for traffic generation/analysis. Version 16 is the most currently released version which shows a significant improvement in throughput performance as compared to previous versions.

2.4.2.2 WARPLab Reference Design

WARPLab is a framework that brings together WARP and MATLAB. With WARPLab, one can interact with WARP nodes directly from the MATLAB workspace and signals generated in MATLAB can be transmitted in real-time over-the-air using WARP nodes. The users can implement different physical layer algorithms in MATLAB which offers the ease for parameter tuning and modification. At the same time, real-time over-the-air transmission/reception performance can be observed through the WARP nodes. This facilitates rapid prototyping of physical layer (PHY) algorithms.

The WARPLab setup is shown in the following figure.

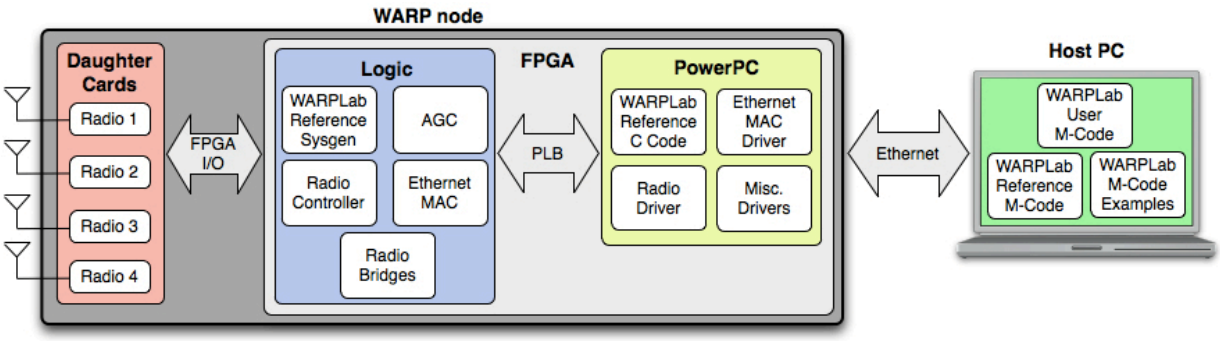


Figure 2-13 WARPLab setup [42]

The WARPLab Reference Design is a Full project that implements the WARPLab design flow. It is basic buffer-based project with no physical or MAC layer. Transmit and receive buffers that take samples from the Ethernet and transmit them over the air are included.

2.4.3 Toolchain for RUn-tiMe Protocol realization (TRUMP)

Based on the custom modifications of the OFDM reference design and implementation of a flexible MAC component library [43], RWTH has developed a tool chain for fast prototyping of MAC solutions. We refer our toolchain as TRUMP (Toolchain for RUn-tiMe Protocol realization) [44]. The toolchain includes the following:

1. Wiring Engine which binds the MAC components,
2. meta-language to describe the MAC design,
3. (host) compiler which converts the MAC language to executable code for a particular target platform,
4. interactive Graphical User Interface (GUI) to ease the MAC designing.

The prototype implementation of the toolchain is carried out on WARP boards. The MAC designing process using the TRUMP is shown in the figure below. The 'Drag and Drop' based interactive GUI provides an IDE for the developers to design a MAC in the form of flowcharts, which is converted into the domain specific language and is sent to the target platform (WARP) over a serial bus interface. The host compiler parses the code and maps it

to the MAC components. The Wiring Engine controls the state-machine of the MAC (i.e., coordinates the control and data flow) and executes the MAC components.

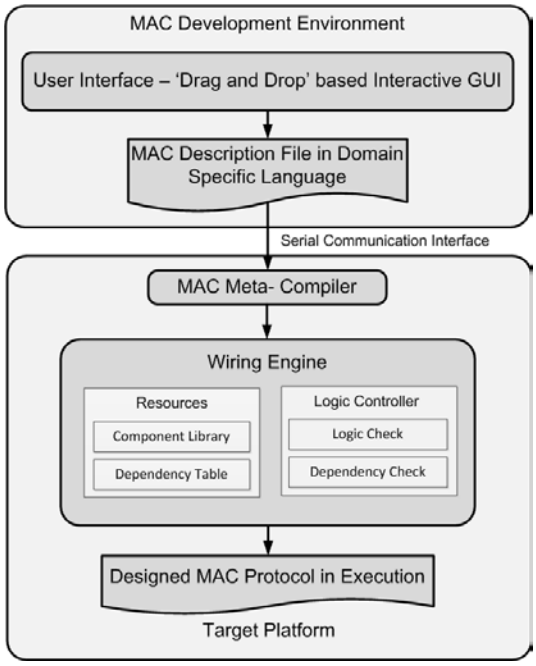


Figure 2-14 Illustration of the MAC realization process using TRUMP [45]

2.4.4 WARP Testbed at RWTH

At the Institute for Networked Systems, RWTH Aachen University, the WARP test-bed is used in the laboratory courses as well as for research activities. The following figure shows a test-bed consisting of different WARP nodes connected to a remote PC for experiments on Cognitive and spectrum agile MAC solutions.

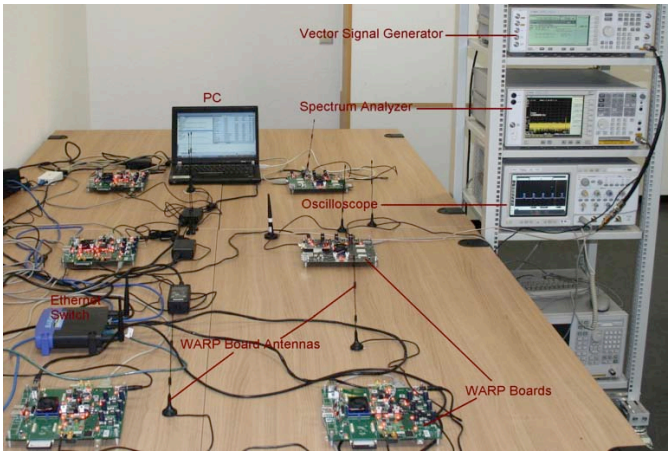


Figure 2-15 WARP testbed used for the evaluation of a spectrum agile Cognitive MAC [47]

Furthermore, RWTH has developed a test-bed consisting of heterogeneous spectrum sensors for fine-grained spectrum sensing in indoor environments. The testbed consists of WARP boards, USRP2 boards and TelosB platforms. The testbed deployment has been carried out across multiple office rooms. The spectrum sensors are connected to embedded PCs and are part of the LAN infrastructure. RWTH has developed driver and library routines,

which allow us to control the sensed bandwidth, switching rates and the sensitivity levels. The testbed allows us to remotely control the tests and logon the data in well-defined formats to be processed and analyzed by statistical tools such as Matlab and R. A deployment scheme of the testbed is shown in the Figure 2-16. Figure 2-17 shows a snapshot of different devices placed on a table as part of the testbed. Overall the largest deployments in RWTH can consist of some 20 WARP boards.

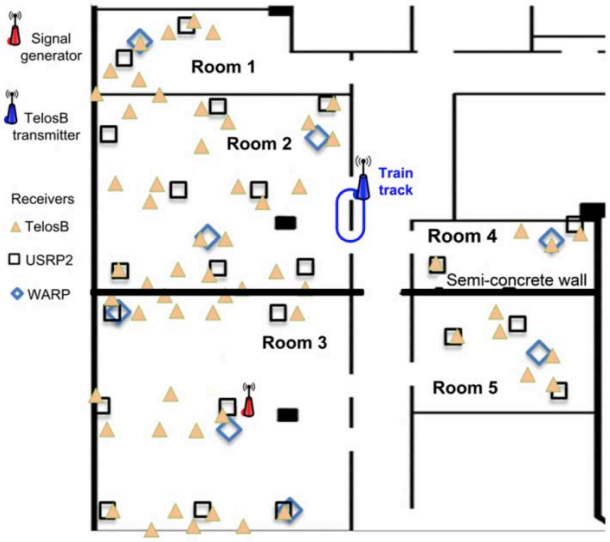


Figure 2-16 Spectrum sensing testbed deployment setup consisting of WARP boards, USRP2 boards and TelosB nodes across different rooms [46]

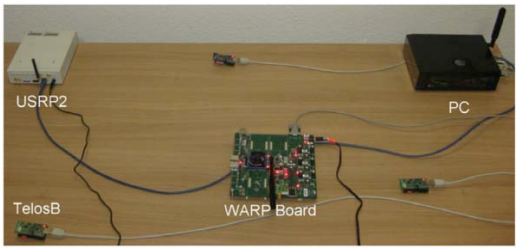


Figure 2-17 Snapshot of the spectrum sensors including a WARP board, USRP2 board and TelosB nodes connected to embedded PC which is part of the LAN infrastructure [46]

2.5 OpenAirInterface platform

2.5.1 Overview

OpenAirInterface provides open-source hardware and software solutions for experimental radio network experimentation. The activity makes use of broadband and spectrally agile hardware platforms, in addition to high-performance emulation software for generic PC computers [49][50].

The software-based platform currently aligns its air-interface development with the evolving LTE standard but provides extensions for mesh networking, particularly in the MAC and Layer 3 protocol stack, in addition to Layer 1 extensions for distributed network synchronization. It can be seen as a mock standard for experimenting with real-time radio resources which retains the salient features of a real radio system, without all the required mechanisms for large-scale network deployment. Networking with tens of nodes using two-way real-time communication in both cellular and mesh topologies has been demonstrated

in the context of several collaborative projects. The aim is to study practical aspects in modern radio systems such as distributed/cooperative processing, distributed synchronization, interference coordination and cancellation, and spectrum aggregation. OpenAirInterface features an open-source software modem written in C comprising physical and link layer functionalities for cellular and mesh network topologies. This software modem can be used either for extensive computer simulations using different channel models or it can be used for real-time operation with the available hardware. In the latter case, it is run under the control of the real-time application interface (RTAI) which is an extension of the Linux operating system.

OpenAirInterface provides a partial implementation of the Rel-8/9 3GPP LTE specifications, primarily related to the access stratum. Specifically we provide

- One or two-antenna transceivers for PHY specifications 36-211/36-212/36-213 corresponding to transmission modes 1, 2 and 6. Modes 4 and 5 are imminent. Currently TD-LTE frame configurations are (partially) implemented as far as 36-213 is concerned.
- Rel-8 MAC layer (36-311), with partial support of random-access and control elements
- Rel-9 RLC (36-321) UM and TM modes. AM is Rel-4 and will be upgraded eventually.

In addition, two different RRC implementations are available. RRC mesh (an ultra-light RRC) which is used for small network deployments and is not 3GPP compliant by any means. It can be used as a testbench for controlling/testing the lower layers. RRC cellular is a subset of 3GPP RRC, but does not currently use ASN.1 encoding/decoding for messages.

Two Linux network devices (nasdriver, nasmesh) are provided to interface the 3GPP stack to Linux. This is a non-3GPP network interconnect. There are plans to integrate OpenAirInterface with open-source 3GPP networking implementations.

OpenAirLTE provides several testing modes, namely:

- unitary Layer 1 simulation of PBCH,PCFICH/PDCCH,PDSCH,PUSCH,PUCCH (coming) for use in basic performance evaluation, PHY abstraction modeling and testing prior to real-time integration on OpenAirInterface hardware. A subset of 3GPP MIMO channel models is provided. These also provide MATLAB/OCTAVE output of the different physical channels (to test compliance of transmitter) and can take textual signal files as input (to test compliance of receiver). These can also be used to feed commercial signal generators with 3GPP-LTE stimulus.
- Full multiuser simulation with Layer 2 protocol stack for system performance evaluation. This is limited to small networks and used to test physical layer procedures and their interfaces with Layer 2.
- Full multiuser emulation (with PHY abstraction) with Layer 2 protocol stack. This mode uses abstraction models for PHY (like commercial system simulators) but can be made to run in real-time on PC-based platforms using the full protocol-stack implementation.

In the following, we will present the OpenAirInterface hardware and software architectures and protocol stack. We will present then the list of capabilities that will be available in each

layer of the protocol stack. In Section 2.5.4, we will present some experiments examples and applications of OpenAirInterface platform.

2.5.2 Hardware/RTOS elements

2.5.2.1 OpenAirInterface CBMIMO1

The CBMIMO1 (Figure 2-18) comprises two time-division duplex RF chains operating at 1.900-1.920 GHz with 5 MHz channels and 21 dBm transmit power per antenna for an OFDM waveform. The cards is mainly responsible for interfacing with RF front-end as well as framing of the transmit and receive signals. Each CBMIMO is equipped with 2 antennas to demonstrate the MIMO operation of WMN.

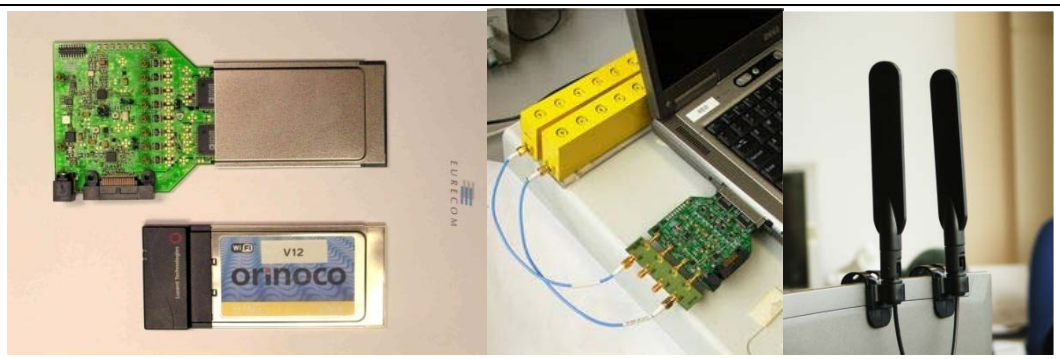


Figure 2-18 OpenAirInterface CBMIMO1 Hardware Components

The CBMIMO card communicates with a host PC over the CardBus/PCMCIA interface. All the PHY layer signal processing is usually run in real-time on the host PC under the control of the real-time application interface3 (RTAI), which is an extension to the Linux operating system.

Figure 2-19 shows the software components and protocol stack of wireless mesh network, which is organized into four folders:

- Openair0 folder mainly contains descriptions of the CBMIMO hardware and the firmware for the corresponding FPGAs. The main parameters to configure are radio frequency, and transmit power.
- Openair1 folder contains the code for the physical layer software modem along with RTAI/Linux device drivers and user-space tools to control the hardware. All the related physical layer parameters are configurable including framing, interleaving, waveform, etc..
- Openair2 folder contains the layer 2 protocol stack development. This pertains to both cellular and mesh network topologies and can be configured according to the application scenarios.
- Openair3 folder contains the layer 3 protocol stack development for both all-IP cellular and mesh networks. In particular, it includes unicast and multicast protocol, topology control procedure, and mobility management.

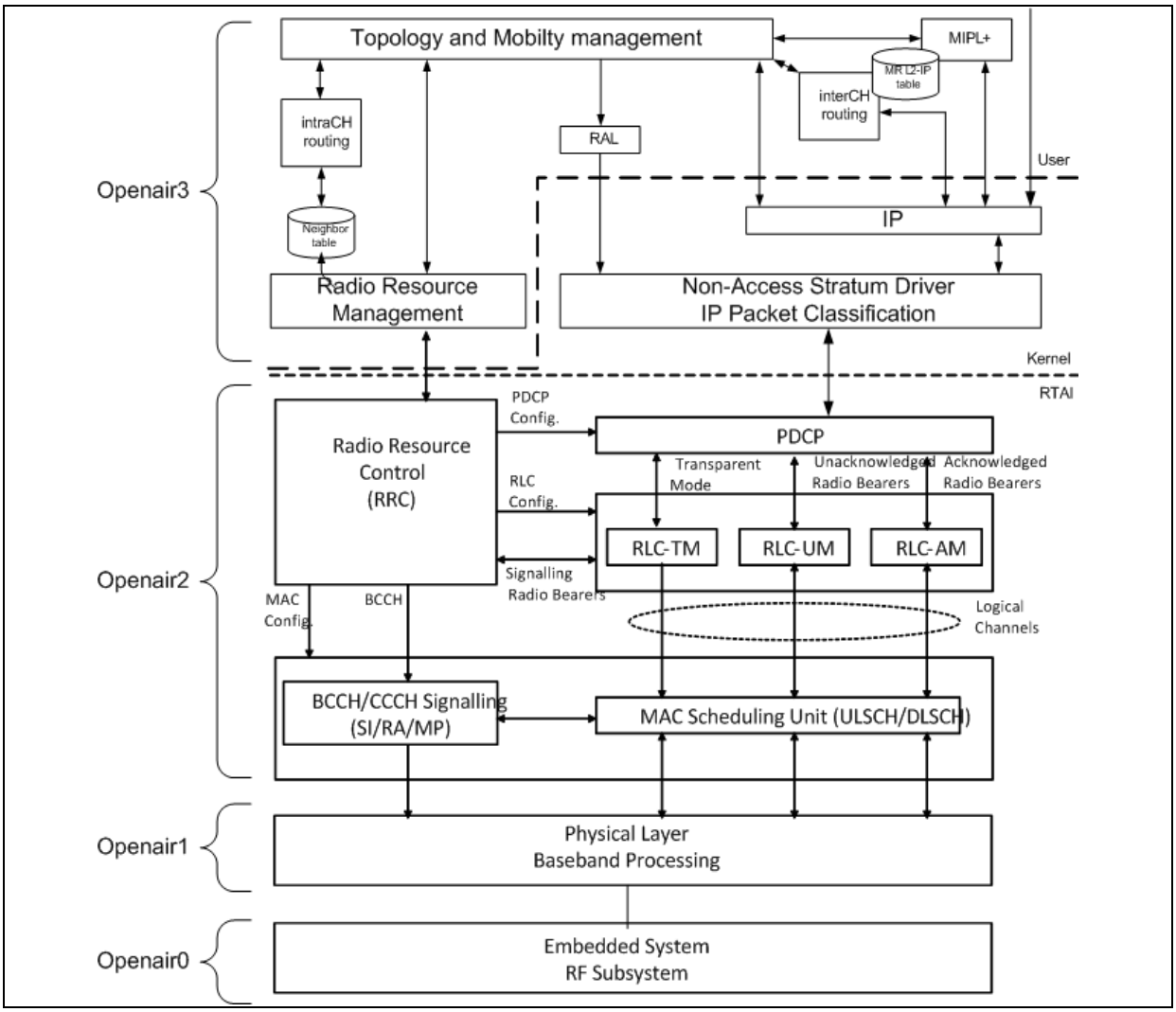


Figure 2-19 OpenAirInterface software components and protocol stack

2.5.2.2 AgileRF Prototype and ExpressMIMO Baseband Engine

EURECOM will provide an SDR solution housing the OpenAirInterface LTE MODEM development. It is based on two HW modules (ExpressMIMO and AgileRF) interfaced with a standard PC as given by Figure 2-20. In this section we will present the AgileRF radio subsystem and in the following we will present the ExpressMIMO Baseband Engine.

Figure 2-20 AgileRF Prototype and ExpressMIMO Baseband Engine

AgileRF is an RF front-end prototype for broadband radio-access. An example configuration is shown in Figure 2-20 consisting of a single TDD transceiver operating over the 150MHz-8GHz frequency range. The AgileRF boards comprise the following subsystems:

- RX: This is a generic broadband receiver board (200 MHz – 8 GHz, 20 MHz channels), Quadrature (I/Q) output
- TX: This is a generic quadrature transmitter board operating in the frequency range of 200 MHz – 8 GHz. RX :
- Synth 1: 8.2 GHz local oscillator (used for systems below 4 GHz, e.g. DAB/DMB, LTE/GSM/WCDMA/HSPA)
- Synth 2 : 4-8 GHz local oscillator

The receiver is comprised of a broadband LNA followed by a band-selection filter network. A direct conversion quadrature mixer is used for inputs in the range 4-8 GHz. An additional upconverter to 4-8 GHz is used for input signals in the 150MHz-4 GHz range. The band-selection filters and RF gain levels are controllable via a digital interface (controlled here by ExpressMIMO). Baseband outputs are provided via differential quadrature (I/Q) signals from the baseband engine. The baseband section has maximal baseband channel bandwidth of 20 MHz and a sharp DC block for RF carrier leakage removal. Baseband amplifiers provide 60 dB of gain, which when combined with variable RF attenuators allow for 70 dB of gain control.

The transmitter has maximal baseband channel bandwidth of 20 MHz. Baseband inputs are provided via differential quadrature (I/Q) signals from the baseband engine. Band-selection filters are provided to guarantee image-free outputs in all target bands. The bands are, DC-200 MHz, 200-400 MHz. 400-600 MHz, 600-1000 MHz, 1-2 GHz, 2-3 GHz, 3-5 GHz, 5-8 GHz.

ExpressMIMO is an 8-way signal processing engine comprising

- One Xilinx Virtex 5 LX110T embedded system
    - 8x PCIexpress
    - 1Gbit/s Ethernet
    - SystemACE Flash
    - 128 kByte DDR memory
    - LVDS expansion
- One Xilinx Virtex 5 LX330 computational engine
    - 2 Gbyte 64-bit DDR2 memory
    - 4 AD9862 Mixed Signal Front-Ends
        - Dual 128 Msamp/s 14-bit D/A
        - Dual 64 Msamp/s 12-bit A/D
        - Auxiliary A/D and D/A
    - 1 AD9510 Precision PLL + VCO programmable clock source
    - Custom RF interface

It is powered by a standard 430W PC ATX power supply. The RF solution is based on a custom broadband RF platform, AgileRFthat covers the 200MHz-8GHz spectrum with channels up to 20MHz. The digital baseband processing platform being developed by Eurecom and TélécomParisTech is generic prototype architecture for SDR applications. Thanks to its multimodal design, almost all existing standards are supported and future

adaptations can be made easily. Simulation and validation is mainly referred to the baseband processing part of the platform. For the sake of completeness, an overview about the considered HW components is given in this section.

The baseband processing takes place between the external radio front-end and the coded samples. As can be seen in Figure 2-21, its functions are split over seven independent IP blocks (hardware accelerators) that can be found on the Processing Engine FPGA (Xilinx Virtex5 LX330):

- Pre-Processor: I/Q multiplexing and basic signal processing functions (sample rate conversion, carrier frequency adjustment,...)
- Frontend Processor (FEP): FFT, IFFT, basic vector operations used for channel estimation, data detection,...
- Mapper & Detector: implementation of different modulation schemes (from BPSK to QAM 256)
- Channel Encoder: convolutional encoding, cyclic block codes
- Channel Decoder: trellis-based decoding algorithms (Viterbi / Turbo decoding)
- Interleaver / Deinterleaver: (de)interleaving of data streams, frame equalization, rate matching operations (repetition, puncturing)

In order to separate processing from control and communication, the IP blocks use the same generic model, which allows an easy upgrading or replacement in the future. This standardized IP shell consists of a memory subsystem (size differs between the blocks), a local microcontroller, the VCIInterface, a DMA engine and the IPCore (contains the specific functionality of each IP block). All blocks are managed by a LEON3 32-bit microprocessor from Gaisler Research, that is part of the Interface & Control FPGA (Xilinx Virtex5 LX110). It is based on the SPARC V8 architecture. Among others, this processor is responsible for the general control and for the scheduling of the whole system. To decrease the amount of traffic on the AVCI crossbar, LEON3 can delegate some specific tasks to the IP blocks (using macros).

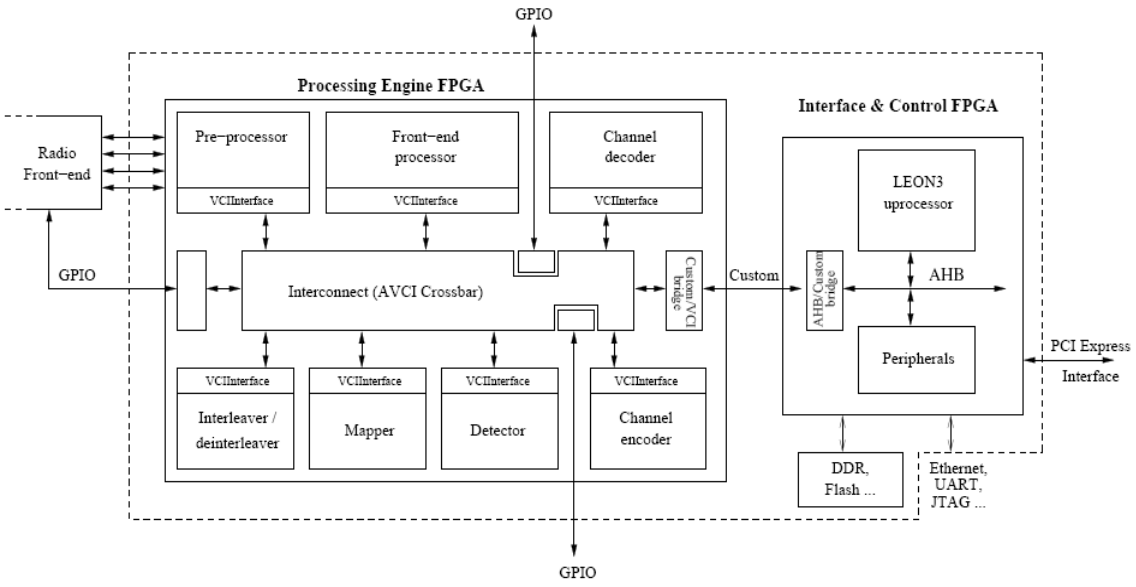


Figure 2-21 ExpressMIMO Baseband Engine

2.5.2.3 ExpressMIMO2 Baseband Engine

ExpressMIMO2 is a baseband platform currently being developed for experimentation with wireless networks. It is meant to be a low-cost solution for broadband reconfigurable radio integrating both FPGA and RF components on a single board and will be available for use at the end of 2011. The characteristics of ExpressMIMO2 are the following:

- Spartan 6 LX150T FPGA (PCIexpress similar to ExpressMIMO)
  - Derived from standard Spartan-6 Xilinx/Avnet evaluation board (but using a smaller, medium-sized PCIe format)
  - Used for FFT and Turbo/Viterbi decoders (key processing bottlenecks)
  - Control of RF and acquisition from converters
- 4 LIME Semiconductor zero-IF RF chipsets
  - TX, RX and A/D, D/A on single-chip (1.5cm x 1.5cm)
  - 300 MHz – 3.8 GHz tuning bandwidth
  - FDD or TDD operation
  - LTE UE, RN RF compliance (EVM), even better (this is really good)
- Additional frequency transposition
  - 4-6 GHz operation
  - 0 dBm output power

ExpressMIMO2 will be interfaced to the same high-power.

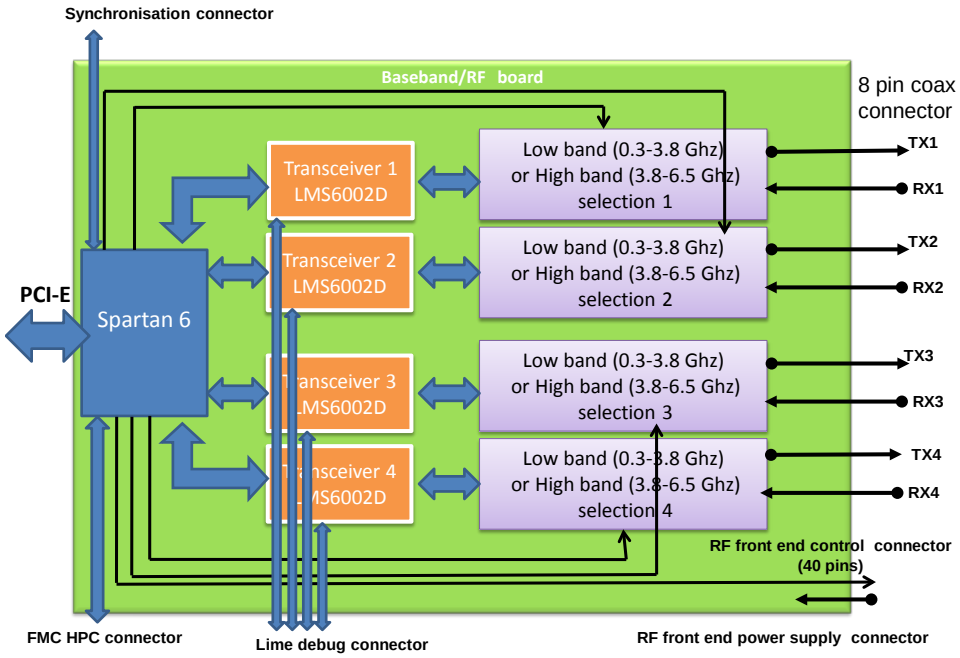


Figure 2-22 ExpressMIMO Baseband Engine Software implementations

2.5.2.4 OpenAirInterface LTE Implementation

The OpenAirInterface initiative recently developed and open-source implementation MODEM implementation for the ExpressMIMO baseband engine and x86 PC targets. This implementation currently provides a standard-compliant LTE Rel-8 implementation of PHY and MAC for a subset of the specifications (36.211,36.212,36.213,36.321,36.322). The gnu-C implementation (with ExpressMIMO or x86 SIMD hardware acceleration) runs under any GNU environment, in particular embedded LEON targets and x86 Linux and RTAI-based targets.

An overview of the currently supported physical/transport channels and transmission modes is given in the following tables. Compliance of the implementation is being validated in conjunction with AT4wireless using their stimulus for the purposes of integration efforts.

Table 2-3 Physical Channel Support in OpenAirInterface (3GPP 36-211)

Physical Channel	**Functionality**	**LTE Compliance**
PSS	TX/RX	Validated
SSS	Not yet implemented	-
Cell-specific Reference signals	TX/RX, Modes 1,2,3,4,5,6 1-2 antenna ports at eNB	Validated for 1 antenna port at eNB
PBCH	TX/RX 1,2 antenna ports at eNB	Validated for 1 antenna port at eNB
PCFICH/PDCCH	TX/RX 1,2 antenna ports at eNB All 5 MHz DCI Formats	Validated for 1 antenna port at eNB, DCI Format 1
PHICH	Partial TX/RX	Not validated yet
PDSCH	TX/RX 1,2 antenna ports at eNB TX Diversity, 2-antenna Precoding (Mode 4/5/6))(	Validated for 1 antenna port at eNB
PUSCH + UCI	TX/RX 1,2 antennas ports at eNB No group/slot hopping	Under validation with AT4
PUCCH	TX/RX formats 1,1a,1b	Under validation with AT4
DRS	TX/RX, 1-2 antenna ports at eNB	Under validation with AT4
SRS	TX/RX, 1-2 antenna ports at eNB	Not validated yet
PRACH	TX/RX, 1-2 antenna ports at eNB	Not validated yet

Table 2-4 Coding and Multiplexing (36.212)

Coding Methods	Functionality	LTE Compliance
Tail-biting C. code, ,	TX/RX	validated
Turbo code	TX/RX	validated
rate-matching (C. code)	TX/RX	validated
Rate-matching (turbo)	TX/RX	validated
segmentation	TX/RX	validated
CRC 24-bit	TX/RX	validated
CRC 16-bit	TX/RX	validated
CRC 8-bit	TX/RX	Not validated
BCH	TX/RX	validated
DCI	TX/RX, 5 MHz formats	Validated (format 1)
DLSCH	TX/RX	Validated
ULSCH/UCI	TX/RX	under validation (subset of UCI formats)
CQI	TX/RX	Under validation
CFI	TX/RX	Validated
HI	TX/RX	Not validated

Table 2-5 Physical Layer Procedures (3GPP 36-213)

Procedure	Functionality	LTE Compliance
Random-Access	TX/RX, simplified to single preamble	Not validated
Random-access response	TX/RX	Not validated
PDCCH procedures	TX/RX	Not validated
DL/UL HARQ procedures	TX/RX, TDD, no PHICH	Not validated
CQI/PMI/RI reporting	TX/RX, HLC and Subband PMI on PUSCH	
PUCCH	Currently being implemented	

2.5.2.5 OpenAirInterface 802.11p Implementation

Another standard that is currently implemented in the context of OpenAirInterface is 802.11p (WLAN in vehicular environments). The physical layer implementation follows the specifications as described in the IEEE Std 802.11™-2007 standard description. CSMA/CA mechanism except ACK, RTS and CTS, is included and the transceiver is intended to be used with broadcast and multicast services.

All required operations of the standard are currently supported by the baseband processing engine of the ExpressMIMO card (see Figure 2-21).

The Preprocessor is used to provide a sample-rate adjustment from the basic 30.72 MS/s sampling rate to the target of 10 MS/s. Besides, Front-end Processor (synchronization, channel estimation, demodulation etc.), Channel decoder / encoder and (De) Interleaver are used. For the DATA field detection, these different hardware accelerators can be processed in parallel.

All blocks are managed by a LEON3 32-bit microprocessor from GaislerResearch that is part of the Interface & Control FPGA. It is based on the SPARC V8 architecture. Among others, this processor is responsible for the general control and for the scheduling of the whole system. To decrease the amount of traffic on the AVCI crossbar, LEON3 can delegate some specific tasks to the microprocessor being part of the different IP blocks. Due to the strong latency requirements, the latter one is strongly recommended for the 802.11p transceiver implementation.

The whole transceiver can be emulated in a pure software environment using the library for ExpressMIMO baseband (libembb). This library contains bit-accurate C/C++ functions representing the behaviour of the baseband processing engine and thus allows a fast verification of the design. In the future it will be possible to use these software stimuli not only for pure software emulation but also for the activation of the real processes on the hardware platform.

2.5.2.6 LTE Protocol stack

The emulation is based on a full real protocol stack implementing a subset of LTE Rel. 8/9 of access stratum (i.e. not modelled) as shown in Figure 2-23 and includes the following blocks:

- Network device driver
- MAC/RLC/PDCP/RRC and IP
- PHY procedures
- Abstraction of physical channels, MODEM is abstracted along with propagation

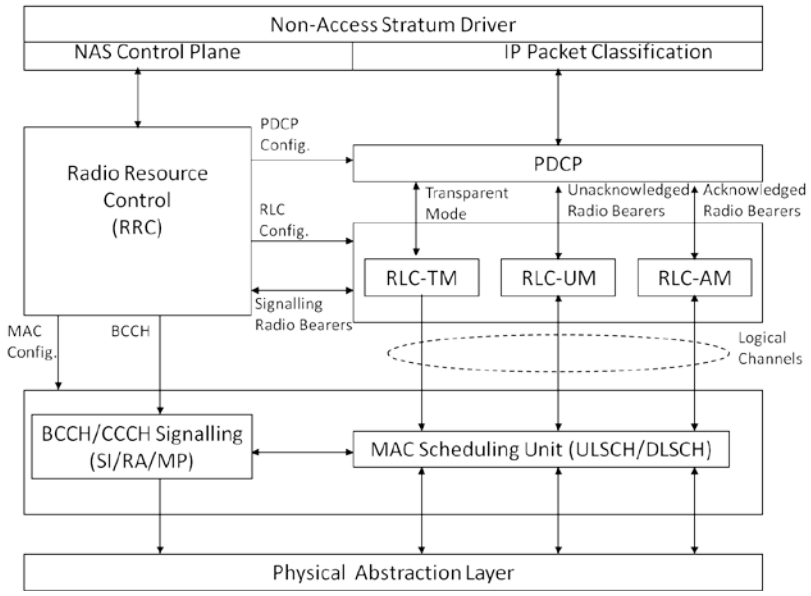


Figure 2-23 OpenAirInterface emu reference protocol stack with phy abstraction

NAS

The direct inter-connection between LTE and IPv6 is performed using an inter-working function, located in the NAS driver and operating in both the Control Plane and the Data Plane. This function provides the middleware for interfacing IPv6-based mechanisms for signalling and user traffic with 3GPP-specific mechanisms for the access network (e.g. for mobility, call admission, etc.). It is developed as an extension of a standard IPv6/IPv4 network device driver.

RRC

The RRC layer, shared between the UE and the ENB, performs the control of the radio interface. It is based on 3GPP 36.331 v9.2.0. The control procedures available in the LTE platform are the following:

- System Information broadcast
- the RRC connection establishment
- the signalling data transfer
- the connection reconfiguration (addition and removal of radio bearers, connection release)
- the paging and the measurement collection and reporting at UE and eNB

These procedures have been extended to support MBMS for multicast and broadcast.

MAC

The MAC layer implements a subset of the 3GPP 36-321 release v8.6 in support of BCH, DLSCH, RACH, and ULSCH channels. The eNB MAC implementation includes:

- RRC interface for SI and CCCH
- Schedulers
- DCI generation
- HARQ Support
- RA procedures and RNTI management

- RLC interface (AM, UM)

UE MAC implementation includes

- PDU formats: timing advance CE, contention resolution CE, BSR CE
- RLC interface (AM,UM)
- RRC interface for SI and CCCH

PDCP

The current PDCP implementation is compatible with Rel 4, and does not support the functionalities with respect to 3GPPP 36-323.

RLC

The RLC layer implements a subset of the 3GPP 36-322 release v8.6 with the following characteristics:

- RLC TM, compatible from rel. 4 to 9 except that for LTE release the segmentation has to be deactivated
- RLC UM
  - rel 6: no support for encryption and decryption. Compatible with UDP-lite
  - rel 9: incompatibility of sequence number (encoded in 10 bits instead of 5 bits)
- RLC AM, compatible with rel. 4
  - Segmentation, concatenation, and reassembly
  - Padding
  - Data transfer to the user
  - Error control and correction

2.5.3 OpenAirInterface Emulator

2.5.3.1 Hardware architecture

The OpenAirInterface emulator makes use of the open-source real-time operating system extension to Linux, RTAI to guarantee hard real-time behaviour. The hardware architecture of OpenAirInterface is shown in Figure 2-24 and makes use of a 64-bit 2GHz Quad-core Xeon cluster with the earlier Linux version 2.6.29.4 (named R4Gx). The clusters are interconnected through a Gigabit switch to enable fast transport of the emulated data traffic. One of the clusters acts as a master and generates periodically the entire emulated parameters for the other clusters. With virtualization of the protocol stack, many instances can reside in the same physical machine.

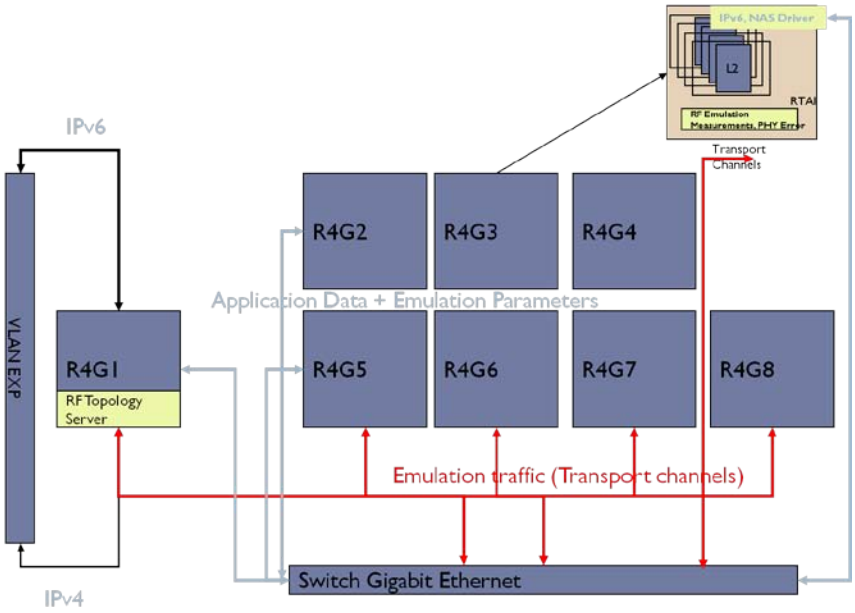


Figure 2-24 Hardware architecture of OpenAirInterface emulation platform

2.5.3.2 Software architecture

Figure 2-25 shows the high level software architecture of the emulated platform itself and the main building blocks. The user scenarios are described using baseline scenario descriptor and then encoded into xml format and dispatched across OpenAirInterface hardware platform according to some predefined rules. Then, the config generator block translates the high level scenarios into low level configuration files understandable for traffic/mobility generator, UE/eNB protocol stack, phy abstraction and emulation transport medium. The real applications are attached on top of some emulated UE/eNB and the remaining traffic pattern and mobility model will be generated automatically. The behaviour of the wireless medium is modelled using a PHY abstraction unit which emulates the error events in the channel decoder and provides emulated measurements from the PHY in real-time. The remainder of the protocol stack for each node instance uses a complete implementation as would a full-RF system. The Log generator is in charge of recording the emulator activities according to the user defined log level, while the packet trace generator captures the frame (potentially extract on-the-fly the relevant field) and anonymizes the payload before storing it. The result generator produces the user defined test results using the outcome of log generator and packet trace generator to the OpenAirInterface portal accessible by the user.

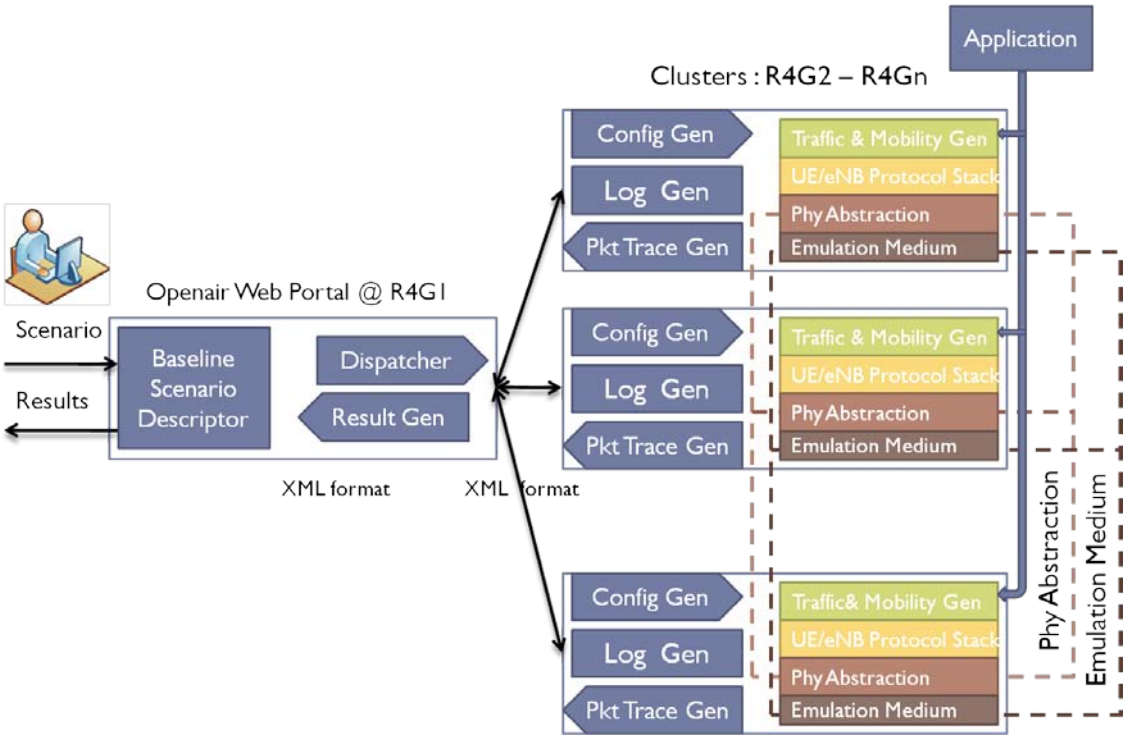


Figure 2-25 Building blocks for OpenAirInterface emulation platform

The OpenAirInterface emulator is designed with the following features in mind:

- allows to emulate a medium-to-large scale networks on one or more machines.
- provides the same conditions that are expected to occur in real-world wireless systems for the network protocols and application.
- accepts real channel measurements or can be based on synthesized channel model
- allows end-to-end IPv4/IPv6 packet transition with QoS support, i.e. any standard application/traffic generator can be attached to any node in the network.
- possibility to interconnect the emulated networks to live networks or to additional emulated network (on-going).

2.5.4 Experiments example

In this section we will present some experiments examples and applications of OpenAirInterface platform.

2.5.4.1 Application of OpenAirMesh platform

One major application of OpenAirMesh platform is the demonstration of rapidly-deployable broadband ad hoc communications systems for public safety units in interventions following natural hazards and industrial accidents. Such a demonstration took place in February 2009 in Bellaterra, Spain in the context of the European project CHORIST, which is funded by the 6th framework programme [97].

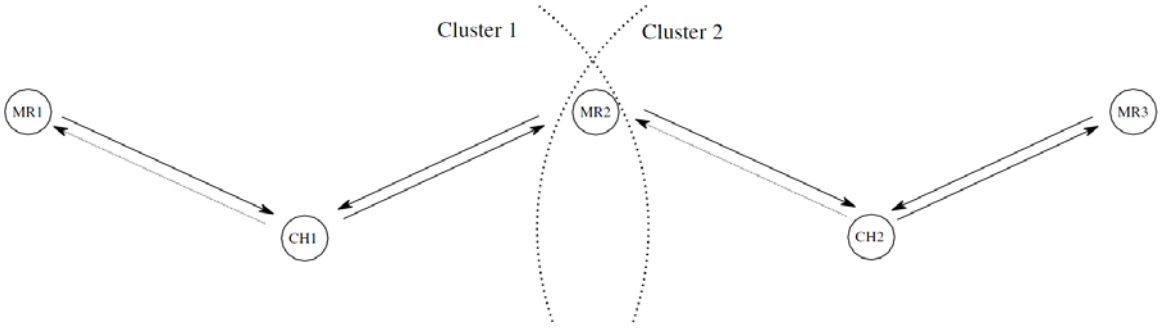


Figure 2-26 The mesh network topology is organized in clusters. Each cluster is controlled by a cluster head (CH). Other nodes in the network are called mesh routers (MR) since they can be used to relay information between CHs.

During the trials we set up a mesh network with five nodes as depicted in Figure 2-26 on the parking lot of the fire brigade building in Bellaterra (see Figure 2-27). MR1 was placed on the roof of the building and served as an edge router establishing the connection to the core network and the control room.

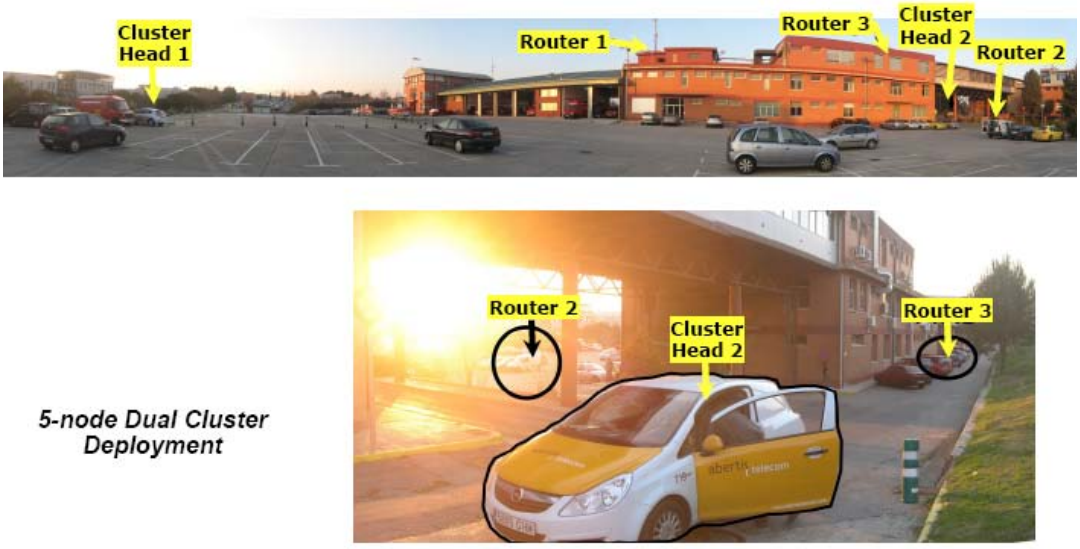


Figure 2-27 5-node Dual Cluster Broadband network deployment

CH1 and MR2 were placed on the parking lot in front of the building. CH2 and MR3 were placed behind the building, such that there was no connection between CH1 and MR3 as well as between CH2 and MR1.Both MR2 and MR3 were used as gateways to other networks. Two different end-to-end applications were tested on the network: a video surveillance application and a push-to-talk VoIP application. During the trials we managed to establish a reliable connection (in the sense that both applications were running smoothly) between MR1 and MR3. For more details on these trials see [97].

2.5.4.2 Signal-level cooperation in cellular network with relaying

The test network has been developed in EURECOM and is based on the OpenAirInterface real-time RF prototype with a PHY allowing experimentation of signal-level cooperation and over-the-air network synchronization. The target networking scenario is an outdoor test-cell facility in Sophia-Antipolis (2.6GHz or 800MHz) comprising 2 eNBs, 2 RSs and 2 UEs. As shown in the figure below, we will first derive a reduced-scale outdoor test environment consisting of 1 eNB, 2 RSs, and 1 UE. This covers the sub-network encircled in blue in Figure 2-28.

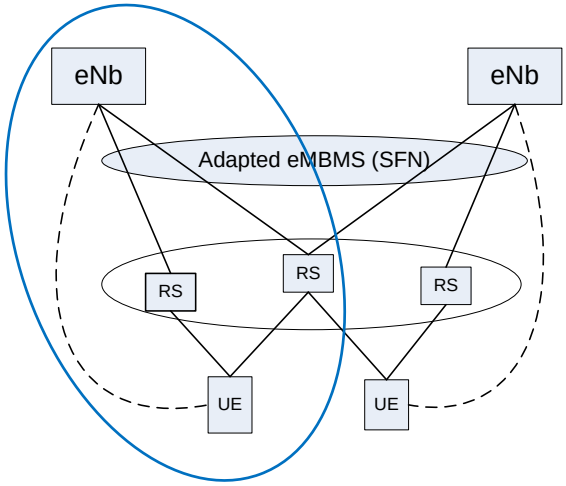


Figure 2-28 Cellular Scenario for LTE-Advanced with relaying

For the links between the RSs and the eNB, we consider a particular channel model: orthogonal error-free limited-capacity backhaul. This represents either a second carrier for eNB to RS communications or the use of orthogonal time slots. The initial deployment will make use of the second method, whereas a second deployment may consider the first as well.

The current setup consists of the eNB running with ExpressMIMO, 4 Lime Semiconductor transceivers (2 sectors, 2 antennas per sector) and the 1.9 GHz TDD front-end (33 dBm, 5 MHz channels) with cross-polarized antennas as shown in Figure 2-19. It has the following characteristics

- 33 dBm output power for 5 /10 MHz bandwidth (compliant with LTE eNB spectral mask)
- TDD duplexing
- RX filtering and amplification for 1900-1920 MHz band

The modules in Figure 2-29 are currently deployed in Sophia-Antipolis at the EURECOM premises for initial experimentation.

Figure 2-29 1.9GHz Antenna System

RS and UE currently use CBMIMO1 (20 dBm) with omni-directional antennas, dual-receivers, single transmitters and are installed in vehicles and powered off the vehicle's engine. The positions of the RS are currently being determined for optimal placement prior to the measurement campaigns. The limitation to 1.9 GHz transmission is due to regulatory constraints in Sophia-Antipolis and the availability of sufficient 1.9 GHz components. The 2.6GHz/800 MHz solutions (10, 20 MHz channels) will be available for deployment in late 2011.

The purpose of the outdoor site is twofold. Firstly, it will be used to perform extensive field measurements in the relay setting which can be subsequently used for the emulation platform (see Section 2.5.3) in order to perform realistic experiments in a controlled setting. Secondly it is meant to be an open-laboratory for partners (both inside and outside the project) to visit Eurecom for joint research. It is currently not envisaged to interconnect the outdoor site to the network to allow for control of experiments from external sources.

2.5.4.3 Two-way relaying for reduced latency

Figure 2-30 displays an LTE-A relaying scenario comprising an eNB a relay-station (RS1) and two UEs. TDD duplexing mode is assumed. This reflects the scenario in an urban environment with eNBs on rooftops and RS in close proximity to UEs, for instance in a tramway or on a streetlamp.

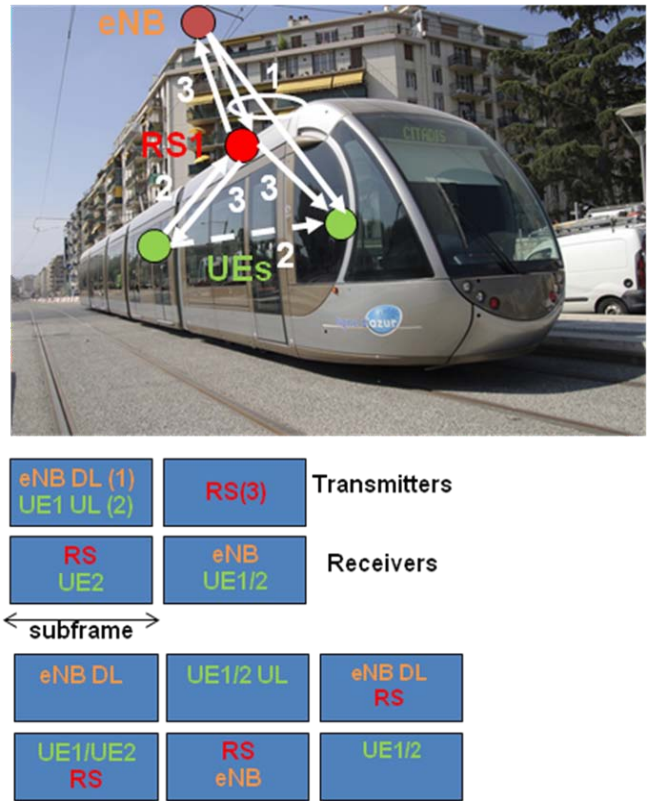


Figure 2-30 Two-way relaying scenario

Here we consider multiplexing of uplink and downlink transmissions at the RS in order to maximize spectral efficiency and clearly lower the latency in comparison to a "classical" relaying approach. The main idea is to employ a two-phase protocol (top framing arrangement) as opposed to a 3-phase protocol (bottom framing arrangement) where uplink traffic from one UE is transmitted concurrently with downlink traffic from the eNB in the first phase. During this phase, the second UE receives the signal from the eNB directly and decodes if possible. In the second phase (in the simplest realization), RS1 transmits the sum signal (amplify-and-forward) which is received by both the eNB and the second UE. The eNB can easily strip its' own signal out of the relayed signal to retrieve the amplified UE transmission. The receiving UE also benefits from two versions of the eNB transmissions during the two phases, if its uplink is inactive during the first phase. The main benefits are

- Two-phase transmission as in normal TDD with the amplification benefits of the relay-station. This will allow for extremely low-power UE transmission (in a Tramway, bus or pedestrian link)
- A 33% latency reduction compared to a three-phase protocol.
- Downlink diversity enhancement (for rightmost UE in the above example) using the two copies of the signal transmitted from eNB and RS.
- Interference mitigation by exploiting interference from the adjacent UE received during the first phase.

We can examine the feasibility of such a technique using the existing LTE/LTE-A waveform with modified physical layer procedures to allow for concurrent transmission of UE and eNB in the case of relay-based deployments.

2.5.4.4 ASIP design for front-end processing

Within a collaboration between the Institute for Communication Technologies and Embedded Systems (ICE, RWTH Aachen) and the Mobile Communications Department (Eurecom), an ASIP for the Front-End Processor IP Core of the OpenAirInterface Platform is designed and compared to the already existing hardware accelerator (custom implementation) [102][103]. The aim of this project is to compare the manual and tool based hardware design approach for multimodal SDR applications and to investigate in the costs for having a flexible front-end processing unit.

Today, air interfaces used by different standards include Orthogonal Frequency Division Multiplexing / Multiple-Access (OFDM/A), Single Carrier FDMA (SC-FDMA), Wideband Code Division Multiple Access (W-CDMA) and Space-Division Multiple Access (SDMA). The set of operations to be performed comprises Channel Estimation, Synchronization, Carrier / Coarse Frequency Offset Estimation or Data Detection. It has been shown that these operations can be build up from a DFT/IDFT unit and a set of different vector operations like component-wise operation or vector sum.

The processing block performing the front-end processing for different standards on the OpenAirInterface platform is called the Front-End Processor (FEP). In the custom implementation based on RTL-level design, the IPCore of the FEP is a mixture between a Vector Operations and a FFT / IDFT unit. For the ASIP design, the IP Core is split into two parts. The FFT unit remains unchanged, while the vector operation unit is replaced by an ASIP.

Both designs support all necessary air-interface operations but on a different level of granularity. While the vector sum, for example, can be processes together with other functions in the custom implementation, it is an own instruction in the ASIP. This difference in the design leads to a different performance when comparing the execution time of the air-interface operations but enable a higher flexibility of the ASIP [102][103].

2.6 CREW federation platform

It is critical to the development of the field to have the ability to implement and test new solutions in cognitive communications, from sensing and spectrum shaping at the physical layer to cooperation and coexistence among cognitive devices in a cognitive network. The *CREW (Cognitive Radio Experimentation World)* federated testbed is the combination of *four* existing cognitive radio testbeds in different European academic institutions, and *two* advanced sensing platforms (Figure 2-31). The testbed includes: the Iris reconfigurable software defined radio (SDR) testbed in Trinity College Dublin (TCD), the TWIST wireless sensor network test environment in TU Berlin (TUB), the w-iLab.t heterogeneous ISM test environment in IBBT, and the EASY-C LTE+ testbed in *TU Dresden (ACROPOLIS partner)*incorporating a complete LTE equivalent base station infrastructure and SDR mobile user terminals. The two advanced sensing platforms are an IMEC (SCALDIO/WARP DIFFS) prototype flexible low power sensing platform, and a THALES multiple receiver antenna array.

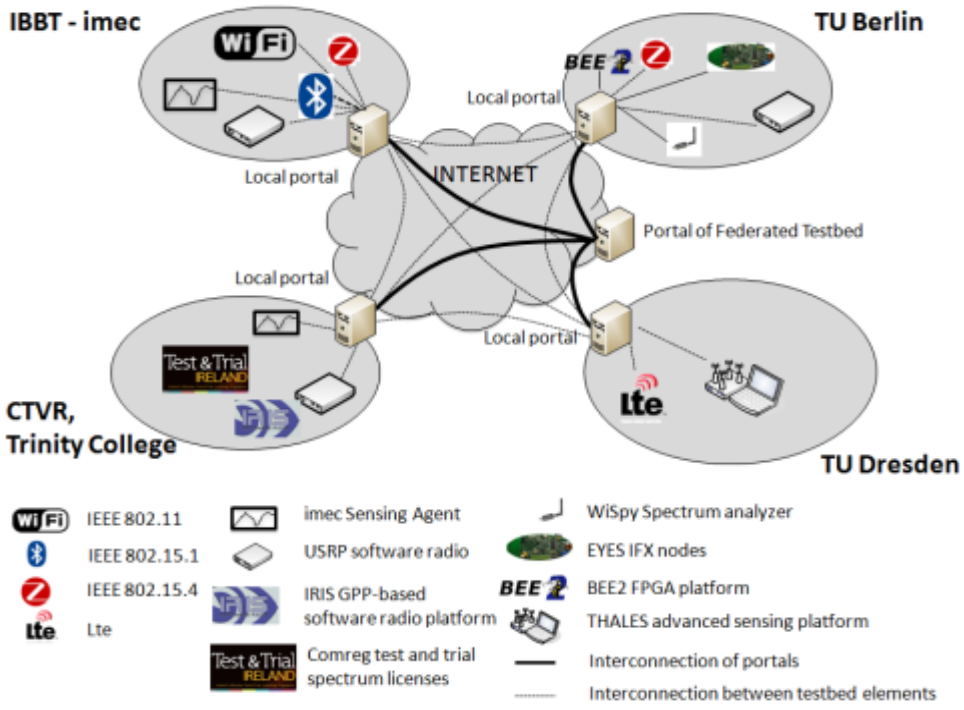


Figure 2-31 The CREW federation platform of CR testbeds [81][85]

2.6.1 Federation testbed key features and functionalities

Table 2-6 shows key features of four wireless testbeds and IMEC advanced spectrum sensing (extracted info based on [85]). For more detailed information on the each of the testbeds can be found in [85].

Table 2-6 CREW federation testbed: key features

Licensed CR testbed **TCD: IRIS reconfigurable radio platform**	**Heterogeneous ISM test environment: w-iLab.t IBBT (Ghent, Belgium)**	**Wireless Sensor Network test environment: TU Berlin TWIST (TKN Wireless Indoor Sensor Network Testbed)**	**LTE cellular test environnent:** **TU Dresden LTE/LTE+ testbed**
Indoor (> 60 sqm)	Indoor 3 Floors (> 3,000 sqm)	Indoor 3 Floors (> 1,500 sqm)	Indoor & outdoor
ISM Band (2.4 GHz, 5.2 GHz) TV – Band	ISM Band (2.4 GHz, 5.2 GHz), IEEE 802.11a/b/g/n IEEE 802.15.1 IEEE 802.15.4	ISM Band (2.4 GHz, 870 MHz)	UMTS Band VII (2.6 GHz)
- *4 Quad core machines* with either a USRP1 or a USRP2 software radio and RFX2400 daughterboard connected to it - *IRIS software radio* platforms (available with either Linux or Windows) - *IMEC sensing engines*	- *> 200* sensor nodes - *~10*imec sensing engines - *~5* Iris software radio platforms - *~5* USRP software radios	- *204* sensor nodes (*102*Tmote Sky and *102*eyesIFXv2) - *~5* Bee2 FPGA platform - *~5* Wi-Spy spectrum analyzer hardware	- *Indoor lab*: Signalion SORBAS platforms as 5 eNB and 5 UEs - *Outdoor lab*: 2 BS sectors, mobile indoor UEs, 3 rickshaw UEs for outdoor exp. - R&S FSH Spectrum Analysers - Real-time and semi real-time exp.
-Remote access to *IRIS* software via *SVN* or *SSH* -the testbed fully accessible via *PuTTy*	Remote access& control (management interface via HTTP, embedded sensor via SSH), OpenVPN is required	- Remote access (via a web-interface)& control; SSH connection to each single node	No remote access and control
IMEC advanced spectrum sensing: • Large range of reconfigurability (Operating freq:100 MHz –6GHz, Bandwidth: 200kHz –40MHz), • Performance, power and area competitive with SotA single-mode radios (Area: 5mm2 (incl. 2 Freq synth, ADC), Power: 40 –100mW, depending on mode), • 'FlexFEC' processor template achieves high speed Turbo & LDPC at low power, • DIFFS - ASIP-based DFE, enables (flexible filtering, synchronization, spectrum sensing)			

The CREW federation functionalities offer:

- *a common portal* to all testbeds, which gives a comprehensive description of the individual testbeds and the functionalities of the federated testbed, including clear guidelines on how to access and use the federated testbed;

- *advanced cognitive components*: spectrum sensing agents and configurable radio platforms, by linking software and hardware solutions from multiple partners using a standardized API;
- *open data sets* for spectrum sensing data and primary user activity, created under benchmarked conditions and using a *common sensing data structure*;
- *a benchmarking framework* for CR and network experiments, which offers automated procedures for experiments and performance evaluation methodologies, comparison between subsequent developments or competing cognitive solutions, and experiments under controlled and reproducible test conditions.

A *three*- mode/step CREW federation roadmap is followed [83] (Figure 2-32)

- In the first mode, *a common portal website* is created with a comprehensive and uniform description of the functionalities and characteristics of each testbed and CR component, access information and usage guidelines.

In this initial step, an access to individual testbeds –initially by all CREW partners, later by the broader public- is enabled;

- The second mode is a heterogeneous usage: physically relocation of hardware and tools. For instance, a software architecture for CR research developed at one location can be installed on top of sensing hardware currently developed at a second location, and then deployed in a controlled wireless experimentation environment at a third location.

In this mode, the individual partner sites remain operational, while the combined expertise and equipment allows more complete and controlled experiments, e.g. a three-testbed HD/SW combination, where every single part of the CR node (radio, network and system stack, testbed control) is fully under control of the experimenters.

- In the third mode, a sequential usage, allows the recording wireless traces in one test environment, and replaying them in other test environments. This possibility is an enabler for repeatable tests and it allows re-creating interference patterns, in the first test location that were recorded in the second test location. Emulation a joint operation of multiple testbeds will be provided through the definition of common data collection and storage mechanism and the provision of subcomponent or subsystem access in each testbed.

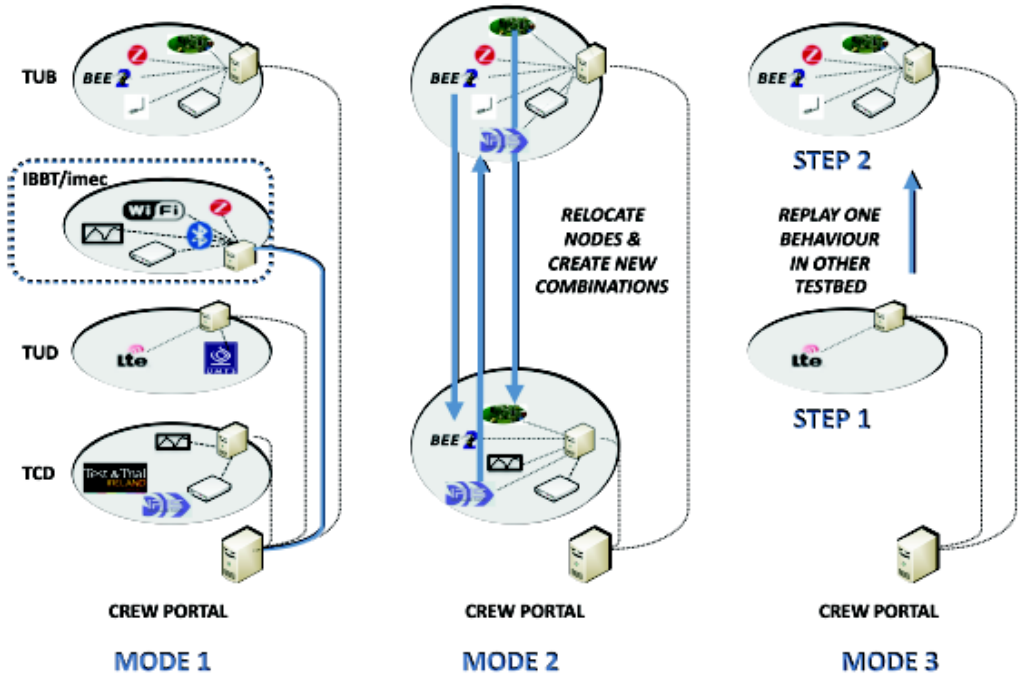


Figure 2-32 Three operating modes in the CREW federation testbed[82]

2.6.2 Usage scenarios

The federation platform allows the experimental validation of CR and CRN concepts for the different usage scenarios (US). Five internal usage scenarios for experimental-driven research have been defined in the CREW project where for each usage scenario a number of use cases (UC) are identified to address particular aspects of relevance in detail.

- (US1) *Context awareness for cognitive networking*: new techniques for context awareness in unlicensed (ISM) and licensed bands (TV white spaces, cellular systems); defined use cases:
    - Context Awareness in the ISM Band
    - Context Awareness in the TV White Spaces Licensed Bands
    - Reliable Sensing of Cellular Systems
- (US2) *Robust cognitive networks*: applications that require robust communications though avoiding harmful interference and using frequency agility to improve communication quality; defined use cases:
    - Cognitive Body Area Networks
    - Cognitive Building Automation
    - Cognitive wireless cabin management system in air planes
- (US3) *Horizontal resource sharing in ISM bands*: algorithms, protocols and networking architectures for coexistence of and cooperation between independent heterogeneous network technologies; sharing in typical home/office/public building environments, densely populated with various wireless ISM band devices; defined use cases:
    - Horizontal resource sharing in home environment

  - Horizontal resource sharing in an office environment
  - Horizontal resource sharing in during an exhibition
- (US4) *Cooperation in heterogeneous networks in licensed bands*: new ideas for opportunistic spectrum access to underutilized licensed bands; defined use cases:
  - Geographical White Space sensing
  - Detection of wireless microphones
  - Transmission and detection in TV bands
- (US5) *Cognitive systems and cellular networks*: the impact of dynamic spectrum access by secondary users on LTE cellular primary systems; sensing agents use in an LTE cellular environment, which operates as primary system; defined use cases:
  - Impact of cognitive networking on cellular primary systems.

2.6.3 Drawbacks of individual testbeds and federation testbed benefits

The **federation** is essential if for those experiments to be carried out successfully, since individual platforms of each partner do not have sufficient capabilities.

The USs require the combination of the high accuracy sensing capabilities provided by the IMEC sensing platform and the on-the-fly radio reconfigurability provided by the *IRIS platform*. IMEC hardware running directly on top of IRIS will noticeable improve the speed of the experiment, cutting out the need for an external node for the sensing equipment and allowing the sensor to exist as a component within the radio.

TUD's testbed was designed for LTE/LTE+ experiments in the scope of the EASY-C project and it contributes a primary cellular system to the federation. It lacks the hardware and algorithms for *spectrum sensing* as well as *reconfigurable radio functionality*, which are both essential for CR and networking specific experiments. By creating the CREW federation, the missing components are introduced to the LTE/LTE+ testbed and a set of experiments are enabled that would not be possible individually.

The *TWIST testbed* (TUB) also does not include any particular cognitive (spectrum sensing) capabilities and the envisioned extensions will integrate low-complexity spectrum sensing devices.

The current *w-iLab.t testbed* (IBBT) does not have advanced capabilities for *RF spectrum scanning*. It means that experimenters designing cognitive networking protocols only have packet and node level information available to help them in analysing the functionality and performance of their solution. The availability of advanced spectrum sensing hardware creates new possibilities.

2.7 Open Access Research Testbed (ORBIT)

ORBIT emulator is a trial network testbed designed to achieve reproducibility of experimentation. It supports also evaluation of protocols and applications in real-world settings. ORBIT (Open Access Research Testbed for Next-Generation Wireless Networks) radio grid testbed was developed for scalable and reproducible evaluation of next-generation wireless network protocols.

The ORBIT testbed consists of an indoor radio grid emulator for controlled experimentation and an outdoor field trial network for end-user evaluations in real-world settings. The radio grid system architecture includes an identification of key hardware and software components. Software design considerations are defined for the open-access radio node, and for the system-level controller that handles management and control. The process of specifying and running experiments on the ORBIT testbed is explained using simple examples in [79].

Currently ORBIT comprises an indoor radio grid of 400 nodes in a 20 m 20 m space, but is planned to be extended into an outdoor field trial network consisting of both high-speed cellular and 802.11 wireless access. A Testbed node in ORBIT is custom-designed, and supports remote management capability through wired connectivity. A user can remotely login and execute experiments. However, since the radio grid is static, topology generation in ORBIT depends on selectively switching some nodes on/off, as in Netbed. However, configuring every link to a desired SNR value is a problem yet to be solved. The mobility of a node is simulated through a separate mobility server, which transfers the state of a mobile node from one node in the grid to another. ORBIT nodes also provide privileged access to a node for performing kernel-related operations. This could taint the kernel running on a node. Hence, disk reimaging at the end of an experiment run is required. Some of the salient features of ORBIT include:

A measurement library that minimizes coding effort of a user by providing predefined functions.

A well designed database storage mechanism of all the results from an experiment, that can be accessed through standard SQL queries.

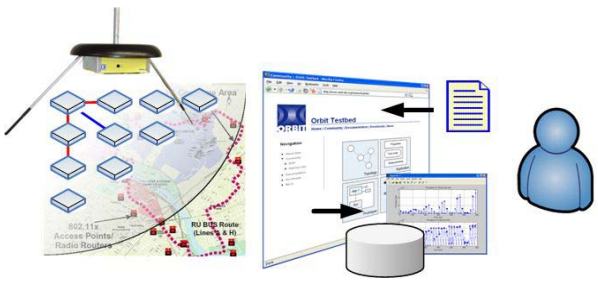


Figure 2-33 Orbit overview [51]

2.7.1 Testbed Overview

The ORBIT Radio Grid Testbed is operated as a shared service to allow a number of projects to conduct wireless network experiments on-site or remotely. Although only one experiment can run on the testbed at a time, automating the use of the testbed allows each one to run quickly, saving the results to a database for later analysis.

In other words, Orbit may be viewed as a set of services into which one inputs an experimental definition and one receives the experimental results as output as illustrated in Figure 2-34below. The experimental definition is a script that interfaces to the ORBIT Services. These services can reboot each of the nodes in the 20x20 grid, then load an operating system, any modified system software and application software on each node, then set the relevant parameters for the experiment in each grid node and in each non-grid node needed to add controlled interference or monitor traffic and interference. The script also specifies the filtering and collection of the experimental data and generates a database schema to support subsequent analysis of that data.

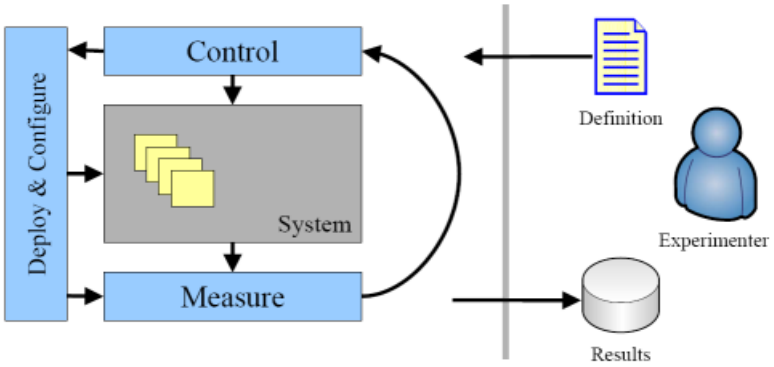


Figure 2-34 Experiment Support Architecture [51]

2.7.2 Hardware Components

The Orbit grid as illustrated in Figure 2-35 consists of a multiply interconnected, 20-by-20 grid of ORBIT Radio Nodes, some non-grid nodes to control R/F spectrum measurements and interference sources, and front-end, application and back-end servers. These servers support various ORBIT services.

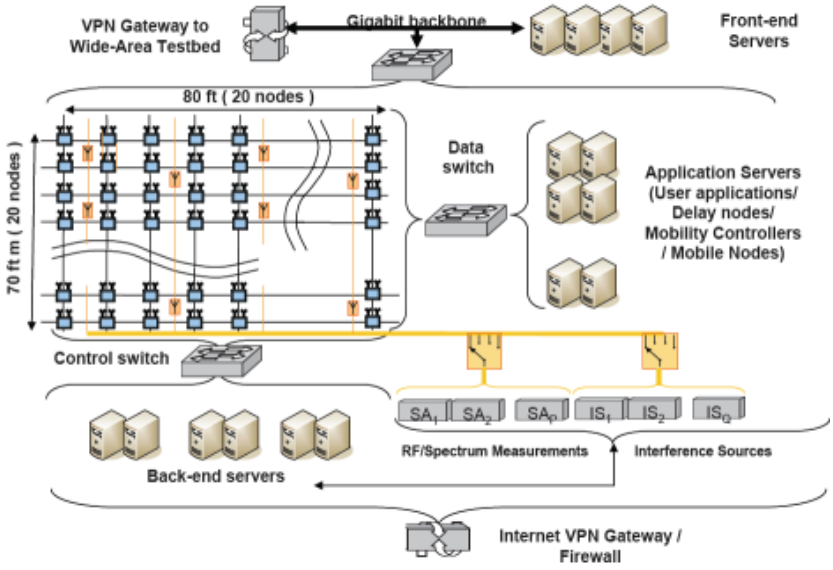


Figure 2-35 ORBIT Hardware [51]

Each ORBIT Radio Node is a PC with a 1 GHz VIA C3 processor, 512 MB of RAM, 20 GB of local disk, two 100BaseT Ethernet ports, two 802.11 a/b/g cards and a Chassis Manager to control the node, see Figure 2-36. The Chassis Manager has a 10BaseT Ethernet port. The two 100BaseT Ethernet ports are for Data and Control. The Data ports are available to the

experimenter. The Control port is used to load and control the ORBIT node and collect measurements.

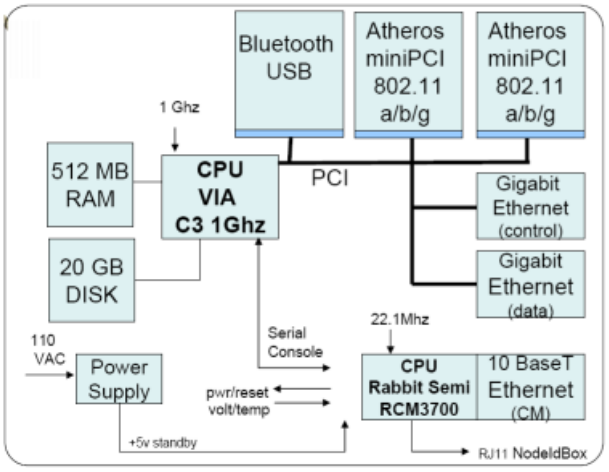


Figure 2-36 Orbit node [51]

2.7.3 Software components

2.7.3.1 Experiment Control

The main component of the Experiment Management Service is the Node Handler that functions as an Experiment Controller. It multicasts commands to the nodes at the appropriate time and keeps track of their execution. The Node Agent software component resides on each node, where it listens and executes the commands from the Node Handler. It also reports information back to the Node Handler. The combination of these two components gives the user the controls over the testbed, and enables the automated collection of experimental results. Because the Node Handler uses a rule-based approach to monitoring and controlling experiments, occasional feedback from experimenters may be required to fine tune its operation. Figure 2-37illustrates the execution of an experiment from the user's point-of-view.

Finally, using the Node Handler (via a dedicated image nodes experiment, which will be described later), the user can quickly load hard disk images onto the nodes of his/her experiment. This imaging process allows different groups of nodes to run different OS images. It relies on a scalable multicast protocol and the operation of a disk-loading Frisbee server from M. Hibler et al.[98]. Similarly, the user can also use the Node Handler save the image of a node's disk into an archive file.

The user can perform all these actions on the testbed(s) via the generic command omf, which is the access point to control the Node Handler, his/her experiment, and the nodes on the testbed(s).

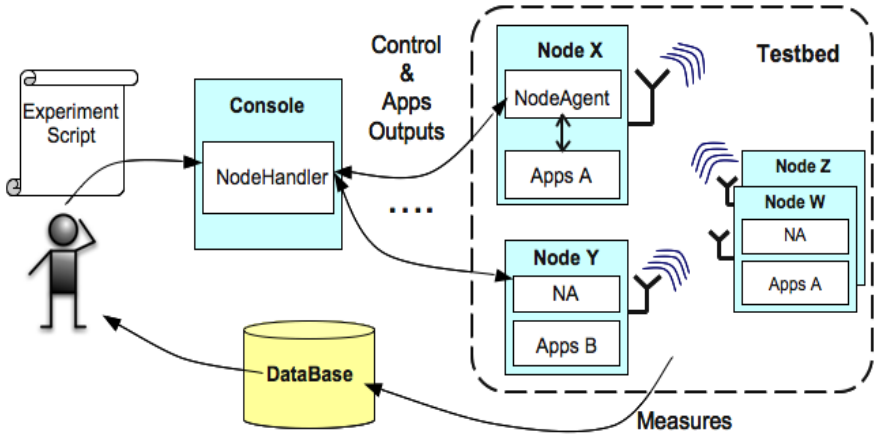


Figure 2-37 Execution of an Experiment from a User's point-of-view [51]

2.7.3.2 Measurement & Result Collection

The ORBIT Measurement Framework & Library (OML) is responsible for collecting the experimental results. It is based on a client/server architecture as illustrated in Figure 2-38 below. One instance of an OML Collection Server is started by the Node Handler for a particular experiment execution. This server will listen and collect experimental results from the various nodes involved in the experiment. It uses an SQL database for persistent data archiving of these results. On each experimental node, one OML Collection Client is associated with each experimental application. The details and "How-To" of such association will be presented in a following part of this tutorial. In the context of this introduction to the testbed, the client-side measurement collection can be viewed as follows. The application will forward any required measurements or outputs to the OML collection client. This OML client will optionally apply some filter/processing to these measurements/outputs, and then sends them to the OML Collection Server (currently over one multicast channel per experiment for logical segregation of data and for scalability). There are two alternative methods for the user to interface their experimental applications with the OML Collection Clients and to define the requested measurement points and parameters. These methods and measurement definitions will be presented in details later in this tutorial.

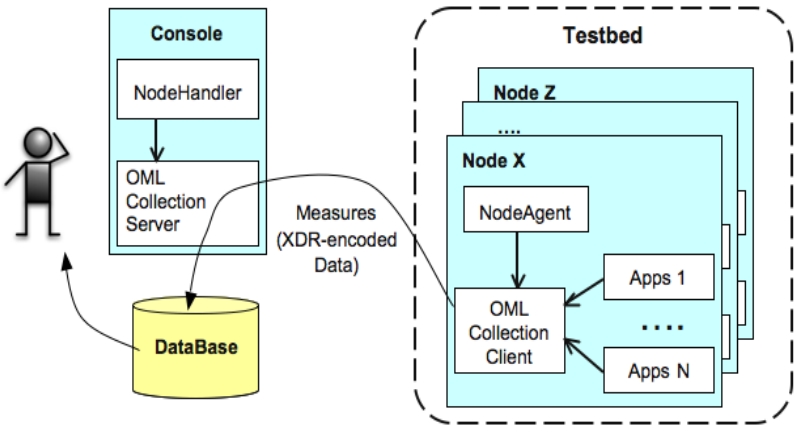


Figure 2-38 OML component architecture [51]

Finally, the ORBIT platform also provides the Libmac library. Libmac is a user-space C library that allows applications to inject and capture MAC layer frames, manipulate wireless

interface parameters at both aggregate and per-frame levels, and communicate wireless interface parameters over the air on a per-frame level. Users can interface their experimental applications with Libmac to collect MAC layer measurements from their experiments.

2.7.3.3 Experimental applications using the ORBIT testbed

ORBIT testbed allows performing a set of experiments involving different network scenarios in the controllable and repeatable environment [99]. The ORBIT software enables to aggregate and analyze the measurements in a simple and efficient way.

Many examples and applications using the ORBIT testbed can be found in [100]. One of these applications will be presented in this section and consist of the investigation of the physical layer capture effect in off-the-shelf 802.11 network cards to confirm that it reduces throughput fairness of traffic flows[101]. All experiments were conducted on the ORBIT testbed comprising64 wireless nodes arranged in an 8x8 grid as shown in Figure 2-39. Each node has two 802.11 a/b/g cards. 802.11b channel 1 is used for all experiments. There is an equal distribution of nodes with Intel IPW 2915 chipset based cards and Atheros AR5212 chipset based cards. For all experiments, the nodes with Atheros cards are used since they allow software control over various parameters such as CWmin selection, disabling retries etc. The open source Madwifi driver for the Atheros chipset based cards implements a majority of MAC protocol features in the driver rather than in hardware, thereby allowing a variety of modifications at the software level. Note that there are no hidden nodes in this testbed and each node is within transmission range of every other node. There is no external interference from other802.11 wireless devices in all the experiments. This was verified by using the iwlist (interface) scan utility that detects infrastructure or ad-hoc networks in the vicinity [101].

Figure 2-39 8x8 radio grid testbed [101]

2.8 Other representative platforms

2.8.1 Kognitiv Networking Over White Spaces (KNOWS)

The Kognitiv Networking Over White Spaces (KNOWS) [57] hardware-software platform has been designed in order to perform unlicensed operations (by unlicensed users – secondary users -SU) in the TV broadcast bands utilizing "white spaces", i.e., portions of the licensed TV bands that are not in active use by incumbent users (primary users -PU) –TV broadcasters. In order to use TV bands such platform must guarantee non-disruptive incumbent reception. Moreover, such systems must have a spectrum-aware Medium Access Control (MAC) protocol, which allows utilizing white-spaces of varying bandwidths. KNOWS detects white-spaces making use of the collaborative sensing, and it deals with the fragmentation of available bandwidth. It employs a *distributed* scheme which dynamically adjusts: the occupancy time, operating frequency, communication bandwidth considering: the instantaneous availability of white-spaces, contention intensity and the user demand.

2.8.1.1 System Overview of the first version of KNOWS

The first architecture of KNOWS [58] is composed of the HD, MAC (spectrum-aware MAC called CMAC), and the spectrum allocation scheme (b-SMART) as depicted Figure 2-40.

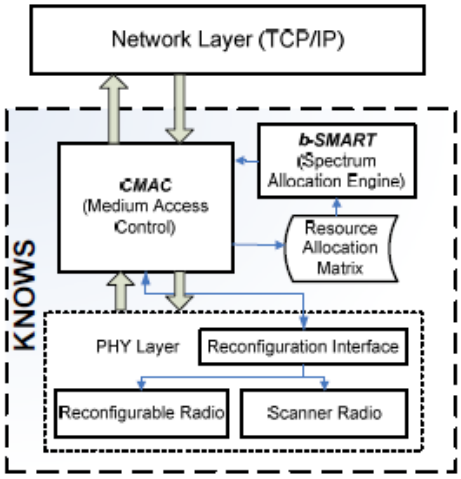


Figure 2-40 KNOWS architecture [58]

Spectrum Aware MAC (CMAC) includes two functions: collaborative sensing –by combining scanning results in one-hop neighbourhood and spectrum reservation scheme using a common control channel (CCC). Nodes contend for the spectrum access on CCC. Upon won contention, a three-way handshake is performed.

b-SMART is an adaptive resource allocation engine and a core component of KNOWS. The objective of b-SMART is to maximize the spectrum utilization while being fair to all users. b-SMART enables at the sender and receiver to collaboratively agreement on a resource block. The reservation is announced on CCC to inform neighbours.

Resource Allocation Matrix (RAM), a local data structure, is used to store spectrum usage information in order to enable efficient spectrum sharing. It records spectrum usage of neighbouring SUs in units of *resource blocks*. A resource block is defined as the time duration and the portion of the spectrum that is reserved by a node for its communication. The bandwidth and time of such block is tuned according to the perceived contention intensity, total available resources and the queue length for each neighbour. The spectrum

allocation engine, b-SMART, utilizes RAM to identify (and assign) the portion of unused spectrum a node should reserve for its communication.

HD includes a dual-mode scanner alternating between a scanner and receiver functions. It scans TV spectrum at least once every 30 minutes during less than 10 ms to scan one 6 MHz TV channel. Working in the receive mode it is tuned to the 902-928 MHz unlicensed ISM band used by a control channel.

2.8.1.2 Hardware of the first version of KNOWS

The HD platform of the first version of KNOWS [58] includes: the reconfigurable radio, scanner radio, GPS receiver and x86 embedded processor. The implementation of this *reconfigurable radio* uses a commodity IEEE 802.11g card in order to generate the OFDM signals at 2.4GHz. A wide band frequency synthesizer is used to convert the received signals to the specified frequency. The interface to the MAC layer is a list of register values in order to control the reconfigurable radio. This list specifies the operating frequency, bandwidth and transmission power level. The frequency can be set from 400 to 928 MHz in 0.5 MHz steps. The bandwidth options are: 5, 10, 20 and 40MHz. The maximum power level is 200 mW. The threshold for packet reception in the TV band is -85 dBm. The time overhead, for adjusting the radio parameters (frequency, bandwidth, power level) is 100 μs in this board.

The *scanner* takes at most 10 ms to scan one 6 MHz TV channel, scanning TV spectrum once every 30 minutes, and most of the time it works as a receiver operating on the control channel. The pilot tone detection sensitivity is -115 dBm.

GPS receiver which performs time synchronization and loads location information, based on the latter identifies the unused spectrum in case a database with TV info is available.

The *x86 embedded processor* controls all radios on the KNOWS platform.

2.8.1.3 System Overview of the second version of KNOWS

The second version of KNOWS has been designed [59], called WhiteFi, focused on the problem of setting up a base station (BS) in the white spaces spectrum. A dedicated control channel is removed in this system, and a new technique (SIFT) is proposed in order to allow nodes for a rapid discovery of BSs operating at different centre frequencies using different channel width (by analyzing signals in time domain). Functional block diagram of the second version of KNOWS is depicted in Figure 2-41.

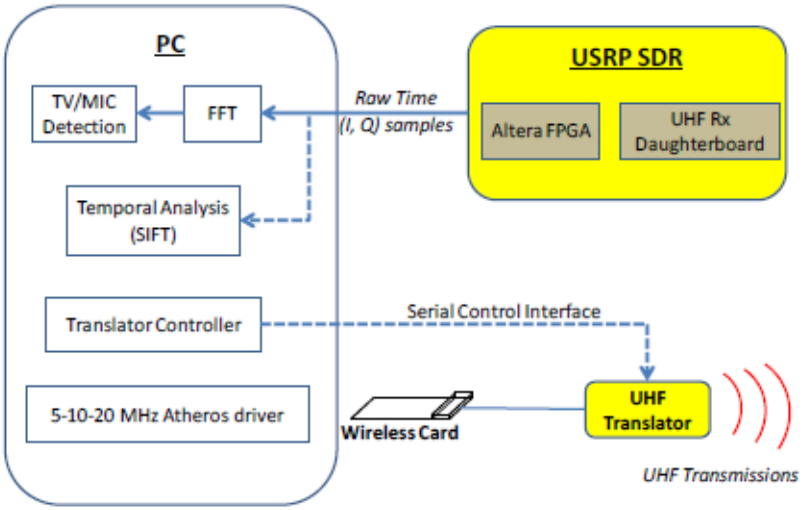


Figure 2-41 KNOWS: Functional block diagram [60]

2.8.1.4 Hardware of the second version of KNOWS

The HD platform of the second version of KNOWS [59] consists of (Figure 2-41):

i. **PC** – equipped with a standard 2.4 GHz Wi-Fi card, antenna port connected to the UHF translator –which downconverts outgoing 2.4GHz signal to 512-698 MHz band.

ii. **UHF translator** – its centre frequency is set from the PC via a serial control interface. Since outgoing signals must fit within a 6 MHz UHF channel, a technique changing the PLL clock frequency [60] is used to reduce the Wi-Fi transmission bandwidth to 5 MHz.

iii. **Scanner** – samples the UHF spectrum in order to detect the presence of TV broadcasts and wireless microphone signals. It uses USRP SDR board coupled with 50-800 MHz TVRX receiver-only dautherboard. It scans 21 -51 UHF TV channels in 6 MHz increments. The frequency span for each scan is 8 MHz due to USRP bandwidth constraints. FFT is performed on the PC, and the feature detection algorithms are used. TV signals at signal strengths as low as -114 dBm and wireless microphones at -110 dBm can be detected (note, that TV decoding threshold is -85 dBm). Hidden Terminal Problem (HTP) can be tackled thanks to this 30 dB detection buffer.

In order to enable efficient networking over white spaces KNOWS2 have extra features (in comparison with the previous version):

i. **Variable Channel Width**: - to support multiple contiguous UHF channels[4] the Atheros Wi-Fi driver is modified[5] in order to transmit and receive signals of bandwidth 5, 10 and 20 MHz.

ii. **Signal Inspection before Fourier Transform (SIFT)**: -which processes raw signals in the time domain and extracts data information from them, this technique reduces the time to detect transmissions in variable channel width systems and allows clients to discover rapidly APs transmitting on a range of channel widths.

2.8.2 Berkeley Emulation Engine (BEE/BEE2)

The Berkeley Emulation Engine (BEE2), developed at Berkeley Wireless Research Center is a generic, multi-purpose platform suitable for for intensive computation [108][109]. The engine consists of five Xilinx Vertex-II VP70 FPGAs connected with high-speed internal links, which provide a possibility to execute in parallel intensive signal processing algorithms. All FPGAs embed PowerPC 405 cores and can be connected to 4 GB of memory with a raw memory throughput of 12.8 Gbps. One of the five FPGAs, called a control FPGA runes Linux OS on its embedded PowerPC to control the peripherals. The remaining FPGAs are used for computation. The BEE2 board supports one 100 Base-T Ethernet which is available on the control FPGA. To interface the BEE2 with radios a total of 18-10 Gbps full-duplex links are available per platform.

The BEE2 can be programmed using Matlab/Simulink from Mathworks coupled with the Xilinx system generator. Due to its modular design and scalability BEE2 is applicable to a wide range of high-performance applications such as real-time radio telescope signal processing, cognitive radios, computer architecture emulation and so on. Due to its high

[4] Existing systems use only one UHF channel (although multiple contiguous UHF channels are unoccupied) due to fixed bandwidth of the outgoing signal to 5 MHz

[5] Techniques presented in [53] are used

cost it has been mainly used for different projects in-house and it has not yet reached to popularity of the USRP and WARP boards.

In the meantime a BEE3 (Berkeley Emulation Engine version 3) has been designed and has been licensed to BEECube [110].

2.8.3 Maynooth Adaptable Radio System (MARS)

The MARS (development in 2004) had the original objectives of a programmable radio front-end that was to be connected to a personal computer (PC) where all the signal processing is implemented on the computers general purpose processor [94]. The platform operates in the frequency 1700 to 2450 MHz. The IRIS framework (See Section2.2) was selected to be a SW framework for an initial development.

Table 2-7 MARS platform release comparison

	MARS (released 2007)	**MARS3 (released 2009)**
Frequency range (MGHz)	1700-2450	1700-2450
RF bandwidth (MHz)	70	25
No. of antennas or RF paths	2	16
Processing partition	Off-board	Mixed
Processor architecture	GPP	FPGA
Connectivity	USB	PCIexpressGigEthernet up to 4 Gbps connectivity
Strengths	Low cost	Large bandwidth
Weaknesses	Limited bandwidth	

2.8.4 Winlab Network Centric Cognitive Radio Platform (WiNC2R)

The WiNC2R [86] is a programmable hardware cognitive radio platform designed by Wireless Information Network laboratory (WINLAB) at Rutgers University. It implements the radio functionality using various HD accelerators programmable by SW. The SW resides in a standard Xilinx soft-core processor (MicorBlaze) implemented on Xilinx Virtex-5 FPGA chip. The HD accelerators are implemented in the FPGA fabric. The platform has been designed for the flexible processing at the radio physical layer and MAC/network layers with sustained bit-rates of *~10* Mbps and higher. The hardware prototype supports multi band operation with a fast spectrum scanning, the ability to dynamically switch between a number of DSSS and OFDM modems and multiple MAC protocols.It operates in 2.4 GHz and 5 GHz ISM bands, 40 MHz RF Bandwidth.

2.8.5 WINLAB GENI CRKit

WINLAB GENI CRKit[87] is a FPGA based embedded platform modular platform with different frontend modules for frequency range from 100 MHz to 7.5 GHz, 25 MHz RF bandwidth. This is an open source platform and application framework based on Matlab/Simulink.

2.8.6 Lyrtech SFF SDR Development Platform

Lyrtech Small Form Factor (SFF) SDR [88] is a FPGA/DSP-based embedded platform with integrated RF frontend. The platform is commercially available. It addresses SDR needs of the commercial markets, public safety and the military.

2.8.7 CoRTekS (CRtestbed using Tektronix Off-the-Shelf Components)

CORTeks (Cognitive Radio Tektronix System)[89] is a cognitive radio testbed using Tektronix test equipment based on two arbitrary waveform generators(AWGs), real-time spectrum analyzer(RSA) and logic analyzer from Tektronix. It is a cognitive engine (CE) implemented using OSSIE open source SCA stack. A neural network-based CE, running on the PC, decides which parameters are best for the given channel state. A neural network provides the flexibility in the decision-making process and allows the radio learn how its decisions impact its goals.

2.9 Other testbeds

2.9.1 Virginia Tech Cognitive Radio Network Testbed (VT-CORNET)

The Virginia Tech Cognitive Radio Network Testbed (CORNET)[90] is a tool to for performing large scale experiments focusing on SDR and CR. CORNET is a large testbed composed by three key elements: (1) SDR nodes with RF hardware based on the USRP2 platform and the Virginia-Tech Cognitive Radio Open Source System (VT-CROSS) framework, (2) a large set of servers characterized by high processing capabilities, and (3) a web interface available for reserving time on the testbed and monitoring the status of the RF platforms.

2.9.1.1 The VT-CROSS framework

The VT-CROSS framework [61] is composed of up to five components, Cognitive Radio Shell, Cognitive Engine (CE), SDR Host Platform, Policy Engine, and Service Management Layer. Figure 2-42, drawn from the documentation available on the CROSS website [61], provides a representation of the interconnections and relations between the components. The different modules interact and communicate by means of standard TCP/IP sockets, allowing for independent development of each component (possibly in different programming languages)

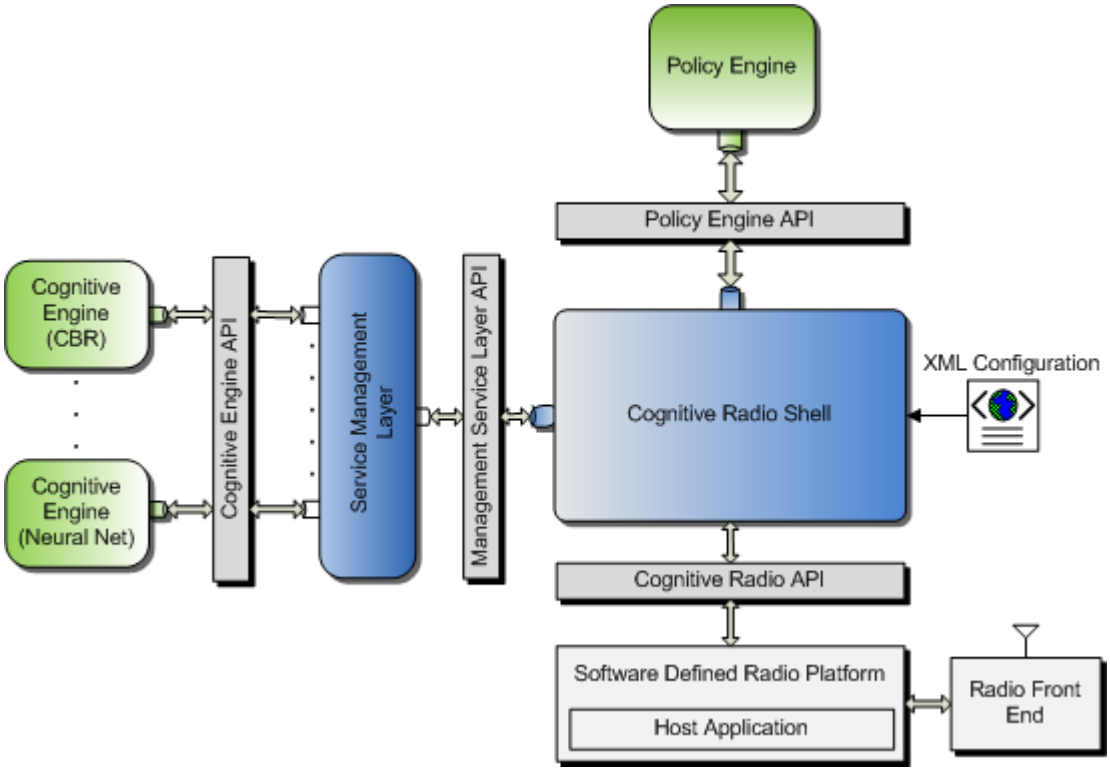


Figure 2-42 The VT-CROSS framework components (drawn from [97])

The *Cognitive Radio Shell (CRS)* has the role of parsing an XML file provided as an input to the framework in order to (1) determine the radio parameters available for configuration on the host platform and their potential values, (2) define the QoS metrics to be taken into account in the operation of the framework, and (3) define the observation variables (e.g.: SNR) and their relationships with the utility functions.

The *Cognitive Engine (CE)* is in charge of deciding how to adapt the behaviour of the cognitive radio, and in particular the settings of the transmission parameters. The CE receives as a first input the available radio parameters and the utility functions parsed by the CRS. During cognitive radio operations, the CE receives observations gathered by the Cognitive Radio Shell as well as any previous experience cached in the CRS, and returns a solution consisting in the best possible settings for the available parameters given the utility function(s).

The *Software-Defined Radio Host Platform* is the actual hardware that gathers observations and emits signals with parameters determine by the CE. In the present implementation of VT-CROSS the platform is based on the USRP2 hardware, complemented by a custom RF daughterboard capable of operations between 100 Mhz and 4 Ghz with 20 MHz instantaneous bandwidth.

The *Policy Engine* has the role of storing and managing policies for spectrum access, and validates the selection of the transmission parameters performed by the CE vs. the currently active policy.

The *Service Management Layer* provides a higher layer interface allowing for the definition of “Missions”, usually involving the optimization of different utility metrics by different instances of the Cognitive Engine. As an example, a Mission could be “Jam all neighbouring devices”, which would involve the sensing of existing devices and the optimization of radio parameters to jam detected devices, possibly adhering to different standards.

Software sources for the VT-CROSS framework can be feely downloaded from the Virginia Tech website. The version now available is still however incomplete in some of its components; as an example, the Cognitive Engine and Policy engine are presently at an early stage of development.

2.9.1.2 Processing servers

The network of servers is composed of 48 servers connected to the Cognitive radio nodes, in charge of performing heavy signal processing tasks that may be required for nodes operation, plus 5 management servers. Servers currently deployed are based on Intel Xeon Quadcore processors and equipped with several GB of RAM [62].

2.9.1.3 Web interface

The web interface provides free access to the testbed (via a registration procedure to create an account) [63]. Users can submit reservation requests that are examined and approved by Virginia Tech system administrators. Reservations can be submitted for the whole testbed or for portions of it (e.g. nodes on one of the 4 floors) allowing for concurrent operations of different users. When a reservation is accepted and the reserved time slot approaches, all nodes are prepared by uploading the OS selected by the user (presently one of several possible Ubuntu Linux versions) along with the selected SDR and CR framework. Presently the following frameworks are available: GNUradio (SDR), OSSIE (SDR), and VT-CROSS (CR). The web interface also allows to monitor the status of each node, and verify whether it is available for experiments.

2.9.2 EMULAB testbed

The Emulab, publicly available testbed from the University of Utah [52], allows conducting a wide range of experimental environments, such as: emulations, live-Internet experimentations (supported by PlanetLab and RON testbeds), IEEE 802.11 Wireless, and Software-Defined Radio (SDR). In order to support SDR experimentations several USRP hardware devices are connected via USB2 to Emulab nodes distributed at the University of Utah. Nodes are equipped with one or more WiFi interfaces. Deployed USRP devices are outfitted with one RFX900 (flex900) 900MHz band transceiver daughterboard (with a directly attached 900MHz "rubber duck" antenna). Using USPR devices gives the control over Layer 1 of a wireless network from signal processing up is done in SW.

An *experiment* is a central operational entity. First, a network topology must be specified in extended the *network simulator (ns)* [53][54] syntax. This virtual topology can include links and LANs with associated characteristics and specifications for hardware and software resources. Once the specification is stored in the database, the “swapin” process to physical resources is executed, where resource allocation is the first step. Physical resources have a complex physical topology, namely, multiple types of PCs with connections via four 100 Mbps or 1000 Mbps Ethernet interfaces to switches connected with multi-gigabit links. The USRP hardware can be used in two ways: As "Bare metal" hardware with nothing setup, and as a wireless network interface with full IP connectivity over a prototype CMSA/CD layer 2 MAC. In order to allocate a node outfitted with a USRP the “usrp” or “flex900” must be specified in the NS file. In order to use GNU Radio USRP hardware as interfaces in an Emulab-managed wireless network experiment a few specific NS directives must be proved in the NS file.

All testbed nodes run a variety of operating systems including FreeBSD, Redhat Linux, Fedora Core, and Windows XP.

2.9.3 WHYNET (Wireless Hybrid Network Testbed)

WHYNET [91], located at the University of California Los Angeles (UCLA), includes an integration of simulation, emulation and physical components for wireless network experimentation. It incorporates 802.11x, 3G cellular and sensor network technology. Beyond the traditional methods of physical experimentation and simulation, it supports several hybrid modes of experimentation (including emulation) that use physical and simulated elements in different combinations. The additional experimentation modes are enabled by a hybrid emulation framework called TWINE [92]that integrates emulation, simulation and physical components seamlessly.

2.9.4 KANSEI

KANSEI, Sensor Testbed for At-Scale Experiments [93], located at Ohio State University, is a heterogeneous testbed with physical hardware components and simulation for sensor network experimentations. It has an access via director framework and it comprises 210 customized static sensor nodes, 50 portable nodes and mobile robot nodes. It uses 433 MHz band, 802.11b, and 802.15.4.It enables high-fidelity experimentation at very large scale through a large-scale simulation with virtual context maintenance of large numbers of nodes.

3. Evaluation and Roadmap

There is no superior platform that would meet all the requirements perfectly. This is quite understandable both from technical and economical point of view. When one is evaluating relative strengths and weaknesses of different platforms following different aspects emerge:

- Research work at Physical Layer and Digital Signal Processing algorithms often require very flexible and powerful platforms. When developing real-time demonstrators and test systems one cannot avoid using FPGAs and dedicated DSP processors which leads to the complexity and higher learning curve of required tools.

- Protocol and networking researchers often require larger number of nodes to test Medium Access Control and higher layers. Although powerful platforms and as modern PHY-layers as possible are preferred the requirement to have “reasonable scale” for experimentation is must. This leads often to the economical limitations, as the very expensive platforms are difficult to buy in large quantities – at least by the university groups.

- Teaching and research requirements are often difficult, but not impossible, to meet. In teaching there is a strong requirement to be cost-efficient (in order to have enough platforms for students), availability of ready-made code and documentation is also imperative. The easy to use tool-chain is also imperative in the teaching that takes in place at the undergraduate level. These requirements tend to favor commercially available and widely used platforms such as USRPs over more expensive and powerful platforms.

- The capability to support different spectrum bands depends on the research problem. Most of the current platforms support well license free ISM bands, but support for other frequency bands is more limited. This generates some hindrance to research in DSA domain, but as many research groups do not have access to experimental licenses to use other band this hindrance is *currently* more theoretical than practical for most ACROPOLIS partners.

- The networking and protocol researchers would often like to build their experiments over plug-and-play PHY-layer, and preferable even MAC layer should be available so that a basic level packet radio network capability can be expected to be ready. This is hardly true for any platforms: some platforms provide reasonable packet radio support, but none have a very good tool chain in this respect. This is one of the largest roadblocks in the current situations, and especially in the case of USRP devices with gnuRadio software this has been commented by many groups as the key problem for their research and teaching.

- Generally the modern transceivers and packet radio systems are highly complex and beyond any single small group to implement easily from scratch. This means that the

importance of the ready-made building blocks, reference systems, and existence of active developer community has become very important. Apart of that many, if not all, platforms have only limited amount of existing and free reference implementations that would allow comparison and testing of the state of the art commercial, cellular systems, such as LTE. The closest platform that provides nearly LTE-type of implementation is open airinterface software and platform.

- The availability, cost and learning curve of different tools is often very important, especially for university groups. There is wide variance on this matter between platforms, and tool-chain situation cannot be deemed to be perfect for any of them. The learning curve of tools, and need to use, is also often deal-breaker, e.g. many pure networking research groups are not inclined to invest money and time to learn complex VHDL and FPGA tools.

3.1 Evaluation of main platforms and roadmaps

In this section we will give short comments and comparative evaluation of key platforms that are used by various ACROPOLIS partners. The consortium conducted a survey over the partners to collect experiences, in order to share main pros and cons related to these platforms. We certainly hope that this helps other research groups on their selection of tools and pushing also various vendors and communities to study possible solutions for shortcoming.

3.1.1 USRP platform

USRP platform, especially with gnuRadio software, is the most commonly used platform within NoE. It seems to be also the most used and the best known in the community in general. Virtually all groups which have acquired USRP-platforms are using them both in teaching and research. The main reported shortcoming or challenges were:

- The lack of professional technical support (manuals, books, data sheets and characteristics of the hardware elements);
- The overall quality of the used elements (such as low selectivity of the interpolation filters, strong nonlinear effects etc.);
- USRP is still at the phase of active development, unfortunately the new drivers are often not backward-compatible;
- It should be possible to shift some of the typical base-band operations (like IFFT in OFDM) could be shifted to FPGA;
- MAC-layer and packet radio support is inadequate, and any published solution for these is either unstable, not publicly available, and/or lack reasonable documentation.
- The need of separate host computer is seen by some groups (especially protocol groups) as problematic for some purpose (see comment below).

Apart of the criticism the USRP platform was highly regarded and widely used. The often referred reasons were an excellent price point that allows its use also in teaching, and relatively large developer community that allows reuse of many components. The key positive comments included:

- The popularity of USRP: the USRP user and developer community is very large; it means that one can find a great number of existing projects, forums, documents and online-discussions;
- The platform is relatively cheap;
- The large variety of daughterboards (dedicated for different application and covering totally very wide frequency band);
- USRP itself host platform independent. Similarly gnuRadio software components can be ported relatively easily, although the main support is for the Linux;
- Various commercial programming platforms (e.g. Matlab, LabView) supports the USRP board, dedicated modules for USRP have been prepared

The roadmap for USRP platform development has become more attractive since National Instruments was acquiring Ettus Research. This has lead already to the announcement of new platform that would have an embedded host computer in the platform itself. The National Instruments seems to be highly committed to develop a platform, and it is to be expected that also manuals, courseware support, and datasheets will start to become ubiquitous.

As a general conclusion, USRP (with or without gnuRadio) was seen as a good cost-efficient platform. It is one of the most ideal ones for flexible teaching, and a wide variety of different research problems can be demonstrated and prototyped with it. However, it still has limitations in tools, and especially if one wants to build the very high-end MIMO capable prototypes.

3.1.2 IRIS

IRIS is a software platform that is developed in Europe. It relies to USRP boards, although the software could be presumably ported to support also other SDR. There was only a limited experience on using IRIS within NoE, the main reported shortcomings were:

- The lack of solid documentation
- Rather small users community
- The number of actually available components is relatively limited
- Still unclear licensing model and thus uncertain availability

Several, partially unique, strengths have been reported for IRIS:

- Independence of the used operating system and type of CPU enabling greater portability;

- Ability to the run-time radio reconfiguration on run-time; various reconfiguration mechanisms;
- Parallel computing capability.

3.1.3 WARP platform

WARP is a very powerful platform that provides good bandwidth and large enough FPGA to synthesize even two RISC-microprocessors cores into it. The platform has gained popularity, and it has relatively steady user and developer base – although not at the level of USRP. The mean challenges and shortcomings related to platform are

- hardware platform is costly especially when compared with USRP2;
- operating frequency limited to 2.4GHz and 5GHz ISM bands;
- Still relatively small developer community;
- Current PHY-layer reference implementation is limited to OFDM.

A number of strong points were reported concerning the platform:

- Good quality open source software, and commercially available hardware platform;
- PHY implemented in the FPGA to enable fast signal processing as well as flexibility;
- Microcontroller(s) can be synthesized into FPGAs to allow standalone use;
- Support for popular and widely used development environment: Xilinx Platform Studio & MATLAB Simulink;
- Good user support from Warp Forum;
- Several hands-on tutorials organized each year, e.g. in Rice University and in May 2011 in DySPAN'11 by Rice University and RWTH Aachen;
- MIMO support up to 2x2.

WARP is still a developing platform with a steady roadmap. New reference implementation refinements and contributions from third parties have been emerging quite regularly for common use. The hardware development has been taken over by Mango Communications, which is currently planning new version(s) of WARP platform. Currently there are four different versions available, the first generation SISO and MIMO board, and the second generation SISO and MIMO kits.

3.1.4 OpenAirInterface Platform

OpenAirInterface has been developed in a number of projects run by EURECOM. It has some unique features and benefits, but its visibility is significantly lower than for USRP or WARP. The main shortcoming of the platform is that it has much more limited user and developer community than its main competitors. As it is not backed by any commercial company its roadmap is also less certain and predictable. One of the strengths is that apart of hardware platform a full simulator and emulator environment is available. The major strengths are

that it has a high performance RF (LTE compliant) and 3G/LTE-type of physical layers have been developed for the platform.

The low cost platform (DBMIMO1) is relatively cheap, fixed band (5 MHz) system, but it offers as hardware only limited competitiveness against USRP or WARP. The new high power versions (ExpressMIMO and AgileRF) can reach the level of WARP boards, but only few devices have been fabricated for internal use only, so far. The current roadmap for the platform is aiming to provide a low-cost platform that would allow also networking experiments in cost-efficient way. The plan is to provide also good quality and flexible frequency agility, although less than in the case of current AgileRF due to cost limitations.

3.1.5 Testbeds

The experience from different existing testbeds has been limited, and results have been somewhat mixed. The most solid experience has been from ORBIT and internal testbeds of partners.

The main attraction and interest on high-scalability testbeds is to get cost efficiently access to large-scale networking capability. The testbeds have been so far mostly interesting for protocol and networking researchers, and in more limited sense there has been interest to use them for various interference management and DSA experiments.

Although the attraction towards testbeds has been initially large the experience has been mixed. The main reported problems were

- Limited high quality manuals and instructions;
- Limited knowledge on the radio environment around the testbeds;
- In most cases quite complex or undocumented tool-chains and configuration tools that are difficult to use without local support;
- Sometimes uncertainty on the availability of the testbed;
- A very long experiments typically discouraged, or not even allowed;
- Possible IPR or confidentiality concerns;
- Even the largest testbeds are quite limited in the size, and some factors like realistic user traffic etc. are nearly impossible to implement.

None of the used or analysed testbeds were emerging as a clear winner, or even being widely particularly popular within NoE or community-at-large.

4. Conclusion

It is clear that none of the platforms are perfect for everyone. In fact, it is doubtful if there ever were a single platform or testbed that could meet all the requirements for different research groups and communities. Currently two hardware platforms, USRP and WARP, are dominating in the research community. In teaching domain USRP is certainly almost the dominant platform in cognitive and cooperative communications community. Both of these platforms are widely used and are still gaining more users. However, one should note that the user community is mostly academic and only a limited number of major companies are using these as R&D tools. The selection between USRP and WARP must be based on the specific project requirements, if the price is not an issue. OpenAirInterface might be a tempting choice if 3G/LTE components are desired and are made available.

One important conclusion from our analysis and user survey is that it is unlikely that any singular project could produce highly competitive platform or testbed unless it makes unexpected quantum leap in technology. Thus funding agencies should carefully analyse any platform projects and try to understand the whole value proposition of platform and testbed proposals. This should not be read as recommendation not to fund such project, but certain realism should prevail in evaluation. The major challenge any new platform is that producing a superior platform is **not** enough. In our survey the requirement to have a good software support, a number of ready mead modules, and excellent documentation was emphasised time and again. The requirement to build a strong user and developer community makes it difficult to establish any rapid changes on the landscape of platforms. Another lesson learned is that there should be increased emphasis and encouragement of different groups and project to share their results, not only through research papers and reports but through shared, free code. ACROPOLIS NoE will certainly study this situation more carefully and seek out what support mechanisms could be provided to encourage such behaviour.

Glossary and Definitions

Acronym	Meaning
ADC	Analogue to Digital Converter
API	Application Program Interface
ASICS	Application Specific Integrated Circuits
BRN	Barrage-Relay Networks
CCA	Clear Channel Assessment
CCM	Configuration Control Module
CE	Cognitive Engine
CellBE	Cell Broadband Engine
CH	Cluster Head
COMM	Communication
CORBA	Common Object Request Broker Architecture
CPC	Cognitive Pilot Channel
CR	Cognitive Radio
CTS	Clear-to-Send
DAC	Digital to Analogue Converter
DSM	Dynamic Spectrum Management
DSONPM	Dynamic, Self-Organising Planning and Management
DSA	Dynamic Spectrum Access
DSP	Digital Signal Processing
EDA	European Defence Agency
ESRA	European Software Radio Architecture
eNB	E Node B
EST	Lower Estimation Loop
ETSI	European Telecommunications Standards Institute
FAR	Flexible/Adaptive/Reconfigurable
FEP	Font-end Processor
FFT	Fast Fourier Transform
FPGAs	Field Programmable Gate Arrays
GPP	General Purpose Processor
GUI	Graphic User Interface
HARQ	Hybrid Automatic Repeat Request
HD	Hardware
IF	Intermediate Frequency
IFFT	Inverse Fast Fourier Transform
ITU	International Telecommunication Union

JTRS	Joint Tactical Radio Systems
JRRM	Joint Radio Resources Management
KNOWS	Kognitiv Networking Over White Spaces
KPIs	Key Performance Indicators
LTE	Long Term Evolution Advanced
LTE-A	Long Term Evolution Advanced
MAC	Medium Access Control
MANETs	Multi-hop mobile networks
MIMO	Multiple Input Multiple Output
MR	Mesh Router
NSD	Noise Spectral Density
OAI	OpenAirInterface
OLA	Opportunistic Large Arrays
ORBIT	Open Access Research Testbed
QoE	Quality-of-Experience
QoS	Quality-of-Service
RAT	2Radio Access Technology
RED	Random Early Detection
REM	Radio Environmental Maps
RF	Radio Frequency
RLC	Radio Link Control
RPC	Remote Procedure Call
RRC	Radio Resource Control
RRS	Reconfigurable Radio Systems
RS	Relay-Station
RSRP	Reference Signal Receive Power
RSSI	Received Signal Strength Indicator
RTAI	Real-Time Application Interface
RTS	Ready-to-Send
SCA	Software Communications Architecture
SDR	Software-Defined Radio
SSH	Secure Shell
SoC	System-on-Chip
SVN	Subversion
SW	Software
TC	Technical Committee
TBF	Token Bucket Flow
TDD	Time Division Duplex

TOA	Time Of Arrival
ToS	Type-of-Service
TR	Technical Report
USPR	Universal Software Radio Peripheral
UE	User Equipment
VO	Vertical Handoff
VRT	VITA Radio Transport
WARP	Wireless open-Access Research Platform
XML	eXtensibleMarkup Language

5. References

[1] J Mitola, "The Software Radio," IEEE National Telesystems Conference, 1992 - Digital Object Identifier 10.1109/NTC.1992.267870.

[2] D. R. Stephens, B. Salisbury, and K. Richardson, “JTRS infrastructure architecture and standards,” in Military Commun. Conf. (MILCOM), pp. 1–5, October 2006.

[3] “The Joint Tactical Radio System (JTRS) and the Army’s Future Combat System (FCS): Issues for Congress” by Andrew Feickert. CRS Report for Congress. Order Code RL33161. November 17, 2005.

[4] PASR 2006 WINTSEC project. http://europa.eu/rapid/pressReleasesAction.do?reference=MEMO/06/375.

[5] FP7 EULER project. http://www.euler-project.eu/.

[6] FCC 08-260: “Second Report and Order and Memorandum Opinion and Order in ET Docket Nos. 02-380 (Additional Spectrum for Unlicensed Devices Below 900 MHz and in the 3 GHz Band) and 04-186 (Unlicensed Operation in the TV Broadcast Bands)”, November 2008.

[7] Harry L. Van Trees, "Detection, Estimation, and Modulation Theory, Part I"John Wiley &Sons, 2001.

[8] Sony, Intel, “TGn Sync TGn Proposal MAC Simulation Methodology”, IEEE 802.11-04/895r2, November 2004.

[9] A. Poloni, S. Valle, “Time Correlated Packet Errors in MAC Simulations”, IEEE Contribution, 802.11-04-0064-00-000n, January 2004.

[10] J. Gilbert et al., “Unified Black Box PHY Abstraction Methodology”, IEEE 802.11- /0218r1, March 2004.

[11] WG5 Evaluation Ad-hoc Group, “1x EV-DV Evaluation Methodology – Addendum (V6),” July 2001.

[12] Ericsson, “System level evaluation of OFDM- further considerations”, TSG-RAN WG1 #35, R1-031303, November 2003.

[13] Nortel, “Effective SIR Computation for OFDM System-Level Simulations,” TSG-RAN WG1 #35, R1-031370, November 2003.

[14] Nortel, “OFDM Exponential Effective SIR Mapping Validation, EESM Simulation Results for System-Level Performance Evaluations,” 3GPP TSG-RAN-1/TSG-RAN-4 Ad-Hoc, R1-040089, January 2004.

[15] K. Brueninghaus et al., “Link performance models for system level simulations of broadband radio access,” IEEE PIMRC, September 2005.

[16] L. Wan, et al, “A fading insensitive performance metric for a unified link quality model” Proceedings of IEEE WCNC, Vol.4, pp. 2110-2114, April 2006.

[17] P.D. Grlinwald, “The Minimum Description Length Principle”, MIT Press 2007.

[18] Mitola, J., III; Maguire, G.Q., Jr., Cognitive Radio: Making Software Radios More Personal", Personal Communications, IEEE, Vol. 6. August 1999.

[19] C. Perkins, E. Belking-Royer, and S. Das, “Ad hoc on-demand distance vector (AODV) routing”, RFC 3561, July 2003.

[20] M. Hassan, and R. Jain, “High performance TCP/IP networking”, Pearson Prentice Hall, 2004.

[21] A.Polydoros, "Algorithmic Aspects of Radio Flexibility", IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC) 2008, Cannes, France, September 2008.

[22] K. Kilkki, "Differentiated Services for the Internet", Macmillan Technical Publishing, Indianapolis, IN, USA, June 1999.

[23] X.Pérez-Costa, D.Camps-Mur and T.Sashihara, “Analysis of the Integration of IEEE 802.11e Capabilities in Battery Limited Mobile Devices”, IEEE Wireless Communications Magazine, Volume 12, Issue 6, December 2005.

[24] J. Mo, J. Walrand and J. Fair, “End-to-End Window-Based Congestion Control”, IEEE/ACM transactions on Networking, vol. 8, pp.556-567, 2000.

[25] R. Jain, D. Chiu, and W. Hawe, "A Quantitative Measure Of Fairness And Discrimination For Resource Allocation In Shared Computer Systems", DEC Research Report TR-301, September 1984.

[26] www.ettus.com

[27] P. Pawelczak, K. Nolan; L. Doyle, SerWah Oh; D. Cabric, "Cognitive radio: Ten years of experimentation and development," *IEEE Communications Magazine*, vol.49, no.3, pp.90-100, March 2011 doi: 10.1109/MCOM.2011.5723805.

[28] S. Koslowski, M. Braun, J. P. Elsner, F. K. Jondral, “Wireless Networks In-the-Loop: Emulating an RF front-end in GNU Radio”, SDR Forum 2010 European Reconfigurable Radio Technologies Workshop, Mainz, Germany, 25 June 2010; available also at: http://www.cel.kit.edu/download/ERRT_2010.pdf.

[29] gnuradio.org

[30] openbts.sourceforge.net

[31] ossie.wireless.vt.edu

[32] Sutton, P.D.; Lotze, J.; Lahlou, H.; Fahmy, S.A.; Nolan, K.E.; Ozgul, B.; Rondeau, T.W.; Noguera, J.; Doyle, L.E.; “Iris: an architecture for cognitive radio networking testbeds”; IEEE Comunications Magazine; September 2010.

[33] Ozgul, B.; Sutton, P.; Doyle, L.; , "Peak power reduction of flexible OFDM waveforms for cognitive radio," New Frontiers in Dynamic Spectrum Access Networks (DySPAN), 2011 IEEE Symposium on , vol., no., pp.664-665, 3-6 May 2011, doi: 10.1109/DYSPAN.2011.5936268

[34] Sutton, P.D.; Lotze, J.; Lahlou, H.; Ozgul, B.; Fahmy, S.A.; Nolan, K.E.; Noguera, J.; Doyle, L.E.; , "Multi-platform demonstrations using the Iris architecture for cognitive radio network testbeds," Cognitive Radio Oriented Wireless Networks & Communications (CROWNCOM), 2010 Proceedings of the Fifth International Conference on , vol., no., pp.1-5, 9-11 June 2010

[35] Sutton, P.; Ozgul, B.; Macaluso, I.; Doyle, L.; , "OFDM Pulse-Shaped Waveforms for Dynamic Spectrum Access Networks," New Frontiers in Dynamic Spectrum, 2010 IEEE Symposium on , vol., no., pp.1-2, 6-9 April 2010

[36] D. Denkovski, V. Atanasovski and L. Gavrilovska, *"Efficient Mid-end Spectrum Sensing Implementation for Cognitive Radio Applications based on USRP2 Devices,"* The First International Conference on Advances in Cognitive Radio (COCORA 2011), Budapest, Hungary, April 17-22, 2011.

[37] E. Blossom, “GNU Radio: Tools for Exploring the RF Spectrum,” Linux Journal, Issue 122, June 2004.

[38] J. Blum, “The Gnuradio Companion (GRC)”. Information available at : http://www.joshknows.com/download/grc_old/grc_gnuradio_hackfest_2009_09_06.pdf.

[39] D. Denkovski, M. Pavloski, V. Atanasovski and L. Gavrilovska, *”Parameter settings for 2.4GHz ISM spectrum measurements,”* 3rd International Workshop on Cognitive Radio and Advanced Spectrum Management (CogArt 2010), Rome, Italy, November 2010.

[40] V. Atanasovski, J. van de Beek, A. Dejonghe, D. Denkovski, L. Gavrilovska, S. Grimoud, P. Mahonen, M. Pavloski, V. Rakovic, J. Riihijarvi and B. Sayrac, *"Constructing Radio Environment Maps with Heterogeneous Spectrum Sensors,"* IEEE Symposia on New Frontiers in Dynamic Spectrum Access Networks (DySPAN) 2011, demonstration, Aachen, Germany, May 2011.

[41] Deliverable D4.1: Radio Environmental Maps: Information Models and Reference Model. EC FP7 FARAMIR project. More information available at: http://www.ict-faramir.eu/index.php?id=deliverables.

[42] WARP-Wireless open-Access Research Platform, Rice University, http://warp.rice.edu

[43] J. Ansari, X. Zhang, A. Achtzehn, M. Petrova and P. Mähönen "A Flexible MAC Development Framework for Cognitive Radio Systems,IEEE WCNC, Cancun, Mexico, 2011.

[44] X. Zhang, J. Ansari, G. Yang and P. Mähönen "TRUMP: Supporting Efficient Realization of Protocols for Cognitive Radio Networks", IEEE DySPAN 2011, Aachen, Germany, May 2011.

[45] J. Ansari, X. Zhang and P. Mähönen "A Compiler Assisted Approach for Component Based MAC Design" In proc. of Med-Hoc-Net, Favignana Island, Sicily, Italy, 2011.

[46] E. Meshkova, J. Ansari, D. Denkovski, J. Riihijärvi, J. Nasreddine, M. Pavloski, L. Gavrilovska, and P. Mähönen "Experimental Spectrum Sensor Testbed for Constructing Indoor Radio Environmental Maps",Poster paper in IEEE DySPAN, Aachen, Germany, May 2011.

[47] J. Ansari, X. Zhang and P. Mähönen "A Decentralized MAC for Opportunistic Spectrum Access in Cognitive Wireless Networks", the 2nd ACM SIGMOBILE Workshop on Cognitive Wireless Networking (CoRoNet) in conj. with MobiCom Illinois, Chicago, USA, 2010.

[48] P. Murphy and A. Sabharwal and B. Aazhang, "Design of WARP: A Flexible Wireless Open-Access Research Platform", EUSIPCO, 2006.

[49] F. Kaltenberger, R. Ghaffar, R. Knopp, H. Anouar, and C. Bonnet, “Design and Implementation of a Single-frequency Mesh Network using OpenAirInterface”, EURASIP Journal on Wireless Communications and Networking, Vol. 2010, Article ID 719523, 16 pages, 2010.

[50] H. Anouar, C. Bonnet, D. Camara, F. Filali, and R. Knopp, “OpenAirInterface simulation platform”, ACM SIGMETRICS Performance Evaluation Review, Vol. 36, N. 2, pp. 90-94, September 2008.

[51] ORBIT two-tier laboratory emulator/field trial network: http://www.orbit-lab.org/

[52] Emulabtestbed:http://www.emulab.net

[53] L. Breslau, D. Estrin, K. Fall, S. Floyd, J. Heidemann, A. Helmy, P. Huang, S. McCanne, K. Varadhan, Y. Xu, and H. Yu. Advances in Network Simulation. 33(5):59-67, May 2000. (An expanded version is available as USC CSD TR 99-702b.).

[54] The Network Simulator –ns-2: http://www.isi.edu/nsnam/ns/

[55] D. Raychaudhuri, I. Seskar, M. Ott, S. Ganu, K. Ramachandran, H. Kremo, R. Siracusa, H. Liu and M. Singh, "Overview of the ORBIT Radio Grid Testbed for Evaluation of Next-Generation Wireless Network Protocols", IEEE Wireless Communications and Networking Conference (WCNC), 2005.

[56] G. Bhanage, I. Seskar, Y. Zhang, D. Raychaudhuri, and S. Jain, "Experimental Evaluation Of OpenVZ From A Testbed Deployment Perspective", 6th international conference of Testbeds and Research Infrastructure (ICST Tridentcom), Berlin, May, 2010.

[57] Networking over White Spaces (KNOWS): http://research.microsoft.com/en-us/projects/KNOWS/

[58] Yuan Yuan, ParamvirBahl, Ranveer Chandra, Philip A. Chou, Ian Ferrell, Thomas Moscibroda, SrihariNarlanka, and Yunnan Wu, “KNOWS: Kognitiv Networking Over White Spaces”, IEEE Dynamic Spectrum Access Networks (DySPAN), IEEE Communications Society, April 2007.

[59] ParamvirBahl, Ranveer Chandra, Thomas Moscibroda, RohanMurty, and Matt Welsh, “White Space Networking with Wi-Fi like Connectivity”, ACM SIGCOMM (Best Paper Award), Association for Computing Machinery, Inc., August 2009.

[60] R. Chandra, R. Mahajan, T. Moscibroda, R. Raghavendra and P. Bahl, ”A case for adaptive channel width in wireless networks”, SIGCOMM, 2008.

[61] Virginia Tech page on the VT-CROSS framework, available at http://cornet.wireless.vt.edu/trac/wiki/Cross

[62] S.M. Hasan, T.R. Newman, B. Hilburn and T. Bose, “Cognitive Radio Network Testbed (CORNET) at Virginia Tech”, talk given at the WICAT Research Review on April 29, 2010. Available online at: http://wireless.vt.edu/research/Cognitive_Radios_Networks/Presentations/Hasan_WICAT_apr_2010.pdf

[63] VT-CORNET web interface: http://cornet.wireless.vt.edu/trac/wiki/CORNET

[64] European Telecommunications Standards Institute (ETSI), http://www.etsi.org, accessed May 2011.

[65] ETSI Reconfigurable Radio Systems (RRS), http://www.etsi.org/website/technologies/RRS.aspx, accessed May 2011.

[66] IEEE Standards Association, http://standards.ieee.org, accessed May 2011.

[67] IEEE 802 LAN/MAN Standards Committee, http://www.ieee802.org, accessed May 2011.

[68] IEEE DySPAN Standards Committee, http://www.dyspan-sc.org, accessed May 2011.

[69] ECMA International, http://www.ecma-international.org, accessed May 2011.

[70] Standard ECMA-392: "MAC and PHY for Operation in TV White Space," http://www.ecma-international.org/publications/standards/Ecma-392.htm, accessed May 2011.

[71] Internet Engineering Task Force (IETF), www.ietf.org, accessed May 2011.

[72] IETF "Protocol to Access White Space database (PAWS)," https://www.ietf.org/mailman/listinfo/paws, accessed May 2011.

[73] IEEE 802.22 Wireless Regional Area Networks working group, http://www.ieee802.org/22, accessed May 2011.

[74] Federal Communications Commission (FCC), Second Memorandum Opinion and Order, in the matter of Unlicensed Operation in the TV Broadcast Bands, September 2010, accessible at http://transition.fcc.gov/Daily_Releases/Daily_Business/2010/db0924/FCC-10-174A1.pdf, accessed May 2011.

[75] IEEE 802.16 Wireless Metropolitan Area Networks working group, http://www.ieee802.org/16, accessed May 2011.

[76] C. Stevenson, G. Chouinard, L. Zhongding, H. Wendong Hu, S. Shellhammer, W. Caldwell, "IEEE 802.22: The first cognitive radio wireless regional area network standard," IEEE Communications Magazine, Vol. 47, No. 1, January 2009.

[77] FCC 08-260: "Second Report and Order and Memorandum Opinion and Order in ET Docket Nos. 02-380 (Additional Spectrum for Unlicensed Devices Below 900 MHz and in the 3 GHz Band) and 04-186 (Unlicensed Operation in the TV Broadcast Bands)", Nov. 2008.

[78] Ofcom, Statement on Cognitive Access to Interleaved Spectrum, 1 July 2009.

[79] D. Raychaudhuri, I. Seskar, M. Ott, S. Ganu, K. Ramachandran, H. Kremo, R. Siracusa, H. Liu, M. Singh, "Overview of the ORBIT radio grid testbed for evaluation of next-generation wireless network protocols" Proc. of the IEEE Wireless Communications and Networking Conference (WCNC), 2005.

[80] GNU Radio website: http://gnuradio.org

[81] CREW project website: http://www.crew-project.eu

[82] The Cognitive Radio Experimentation World (CREW) presentation for the 1st CPqD Workshop on Cognitive Radio and Dynamic Spectrum Access.

[83] Moerman, I., S. Bouckaert, D. Willkomm, J. - H. Hauer, N. Michailow, G. Fettweis, L. A. DaSilva, J. Tallon, S. Pollin, and A. Sanchez, "Testbed Federation: An Approach for Experimentation-Driven Research in Cognitive Radios and Cognitive Networking", Future Network & Mobile Summit, Warsaw, Poland, 15-17 June 2011.

[84] CREW Public Document: D2.1 Definition of Internal Usage Scenarios (January 2011).

[85] CREW Public Document: D2.2 Definition of Federation Functionality (March 2011).

[86] WINLAB WINC2R (Network Centric Cognitive Radio Platform) website: http://www.winlab.rutgers.edu/docs/focus/WiNC2R.html

[87] WINLAB GENI CRKitwebsite: http://groups.geni.net/geni/wiki/COGRADIO

[88] Lyrtech SFF SDR Development Platform website: http://www.lyrtech.com/Products/SFF_SDR_development_platforms.php

[89] CoRTekS (Cognitive Radio test bed using Tektronix Off-the-Shelf Components) website: http://ossie.wireless.vt.edu/trac/wiki/CORTEKS

[90] CORNET (Cognitive Radio Network Testbed) website: http://wireless.vt.edu/research/Cognitive_Radios_Networks/

[91] WHYNET (Wireless Hybrid Network Testbed) website: http://www.cs.ucla.edu/

[92] J. Zhou, Z. Ji, and R. Bagrodia. TWINE: A Hybrid Emulation Testbed for Wireless Networks and Applications. Proc. IEEE Infocom, 2006.

[93] KANSEI website: http://cast.cse.ohio-state.edu/kansei/

[94] R. Farrell, M.Sanchez, and G.Corley, Software-Defined Radio Demonstrators: an example and future trends, International Journal of Digital Multimedia Broadcasting, Article ID 547650, Vol. 2009.

[95] ETSI TR 102 680. SDR Reference Architecture for Mobile Device. Available at www.etsi.org.

[96] ETSI TR 102 839. Mobile Device Architecture and Services.Available at www.etsi.org.

[97] CHORIST website: www.chorist.eu

[98] M. Hibler, L. Stoller, J. Lepreau, R. Ricci and C. Barb, Fast, Scalable Disk Imaging with Frisbee, http://www.cs.utah.edu/flux/papers/frisbee-usenix03-base.html

[99] S. Ganu, H. Kremo, R. Howard, and I. Seskar, “Addressing repeatability in wireless experiments using ORBIT testbed,” First International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities, Trento, Italy, February 2005. [Online]. Available: http://www.orbitlab.org/download/publications/Orbit Repeatability.pdf

[100] ORBIT website: http://www.orbit-lab.org/

[101] S.Ganu, K.Ramachandran, M.Gruteser and I.Seskar, "Methods for Restoring MAC Layer Fairness in IEEE 802.11 Networks with Physical Layer Capture , Proceedings of the Second International Workshop on Multi-hop Ad Hoc Networks, REALMAN 2006, in conjunction with Mobihoc 2006.

[102] C. Schmidt-Knorreck, R. Pacalet, A. Minwegen, U. Deidersen, T. Kempf, R. Knopp, G. Ascheid, “Flexible Front-End Processing for Software-Defined-Radio Applications”, submitted to DATE’12, Design, Automation & Test in Europe, 12-16, March, 2012, Dresden, Germany.

[103] C. Schmidt-Knorreck, R. Pacalet, A. Minwegen, U. Deidersen, T. Kempf, R. Knopp, G. Ascheid, “Application-Specific Instruction-Set Processor for Digital Front-End Processing in Wireless Communications”, submitted toACROPOLIS 1st Annual Workshop on “Advanced coexistence technologies for radio resource usage optimisation”, 4-5, October 2011, Barcelona, Spain.

[104] F.K. Jondral, “Software-Defined Radio –basics and evolution to Cognitive Radio”, EURASIP Journal on Wireless Communications and Networking, Vol.3, pp. 275-283, 2005

[105] S. Haykin, “Fundamental Issues in Cognitive Radio”, McMaster University, Canada, Springer, 2007

[106] I.F. Akyildiz, W.Y. Lee, M.C. Vuran, S.Mohanty, “Next generation/dynamic spectrum access/cognitive radio wireless networks: a survey”, Elsevier Computer Networks Journal, Vol.50, pp.2127-2159, 2006

[107] J. Mitola, “Cognitive Radio: An Integrated Agent Architecture for Software Defined Radio”, PhD thesis, Royal Institute of Technology, Sweden, 2000

[108] C. Chang, J. Wawrzynek, and R. W. Brodersen, “BEE2: A High-End Reconfigurable Computing System,” IEEE Design & Test of Computers, vol. 22, pp. 114–125, June 2005.

[109] http://bee2.eecs.berkeley.edu/

[110] http://www.beecube.com/