Building a Cognitive Radio: From Architecture Definition to Prototype Implementation

Bin Le

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY
in
Electrical Engineering

Dr. C. W. Bostian (Chair)
Dr. R. M. Buehrer
Dr. M. S. Hsiao
Dr. A. B. MacKenzie
Dr. J. H. Reed
Dr. T. Smith-Jackson

11 June 2007
Blacksburg, Virginia

Keywords: Cognitive Radio, Cognitive Engine, Radio Environment Recognition, Machine Learning, Case-Based Reasoning, Genetic Algorithm, Software Defined Radio, Public Safety Interoperability

Building a Cognitive Radio: From Architecture Definition to Prototype Implementation

Bin Le

(Abstract)

Cognitive radio (CR) technology introduces a revolutionary wireless communication mechanism in terminals and network segments, so that they are able to learn their environment and adapt intelligently to the most appropriate way of providing the service for the user’s exact need. By supporting multi-band, mode-mode cognitive applications, the cognitive radio addresses an interactive way of managing the spectrum that harmonizes technology, market and regulation.

This dissertation gives a complete story of building a cognitive radio. It goes through concept clarification, architecture definition, functional block building, system integration, and finally to the implementation of a fully-functional cognitive radio node prototype that can be directly packaged for application use. This dissertation starts with a comprehensive review of CR research from its origin to today. Several fundamental research issues are then addressed to let the reader know what makes CR a challenging and interesting research area. Then the CR system solution is introduced with the details of its hierarchical functional architecture called the Egg Model, modular software system called the cognitive engine, and the kernel machine learning mechanism called the cognition cycle.

Next, this dissertation discusses the design of specific functional building blocks which incorporate environment awareness, solution making, and adaptation. These building blocks are designed to focus on the radio domain that mainly concerns the radio environment and the radio platform. Awareness of the radio environment is achieved by extracting the key environmental features and applying statistical pattern recognition methods including artificial neural networks and k-nearest neighbor clustering. Solutions for the radio behavior are made according to the recognized environment and the previous knowledge through case based reasoning, and further adapted or optimized through genetic algorithm solution search. New experiences are gained through the practice of the new solution, and thus the CR’s knowledge evolves for future use; therefore, the CR’s performance continues improving with this reinforcement learning approach. To deploy the solved solution in terms of the radio’s parameters, a platform independent radio interface is designed. With this general radio interface, the algorithms in the cognitive engine software system can be applied to various radio hardware platforms.

To support and verify designed cognitive algorithms and cognitive functionalities, a complete reconfigurable SDR platform, called the CWT^2 waveform framework, is designed in this dissertation. In this waveform framework, a hierarchical configuration and control system is constructed to support flexible, real-time waveform reconfigurability.

Integrating all the building blocks described above allows a complete CR node system. Based on this general CR node structure, a fully-functional Public Safety Cognitive Radio (PSCR) node is prototyped to provide the universal interoperability for public safety communications. Although the complete PSCR node software system has been packaged to an official release including installation guide and user/developer manuals, the process of building a cognitive radio from concept to a functional prototype is not the end of the CR research; on-going and future research issues are addressed in the last chapter of the dissertation.

Acknowledgements

Dr. Charles Bostian has been more than an advisor to me. His mentorship and friendship has helped me grow professionally and personally. He guided me through various opportunities to learn and contribute to the community. His character and integrity is unparalleled, and I appreciate his steady wisdom, deep understanding, and consistent support. I feel lucky and honored to have been his advisee and worked with him.

I would like to thank Dr. Jeffery H. Reed, Dr. Allen B. MacKenzie, and Dr. R. Michael Buehrer for their previous questions and suggestion during the course of my Ph. D. research. I would like to thank Dr. Michael Hsiao, Dr. Tonya Smith-Jackson for their research guidance and their graduate students in helping me implement the complete system prototype.

I had the pleasure of working with a number of student colleagues in the Center for Wireless Telecommunications (CWT) at Virginia Tech. I appreciate all the brain storming conversations and problem discussions with Tom Rondeau from the first day I joined CWT. I also appreciate all the hard work and spiritual support from the student colleagues in my CWT^2 team, Qinqin Chen, Paco Garcia, Bin Philip Li, Mustafa EI Nainay, and Terry Brisebois. I would also like to acknowledge all the students in our lab and in our project teams for their friendship and technical support.

Finally, thanks to Judy Hood, our lab “queen”, and James Dunson, our lab network administrator.

This research is supported by Award No. 2005-IJ-CX-K017 awarded by the National Institute of Justice, Office of Justice Programs, and US Department of Justice. It is also supported by the National Science Foundation under Grants No. 9983463, DGE-9987586, and CNS-0519959.

For my wife and my family, and any future additions, I love you so very much.

Table of Contents

List of Figures

List of Tables

Chapter 1: Introduction

1.1 Cognitive Radio Research Motivations

The need for flexible and robust wireless communications is becoming more apparent everyday. The future of wireless communications has been anticipated as an evolution and convergence of mobile communication systems and Internet Protocol (IP) technologies, to offer a great variety of innovative services over a multitude of radio access technologies [1]. The most commonly used of these include cellular networks like Global System for Mobile Communications (GSM) and its Generalized Packet Radio Service (GPRS) or Enhanced Data rate for GSM Evolution (EDGE) extension, European Universal Mobile Telecommunications System (UMTS), Wide-band Code-Division Multiple Access (WCDMA), and North America CDMA2000, or Wireless Local Area Networks (WLANs), Worldwide Interoperability for Microwave Access (WiMAX), and Digital Video Broadcasting (DVB). These discrete technologies, each of which contain multiple specific link level standards, are currently transforming to one global wireless access infrastructure, aiming at offering integrated services, according to user demands, in a cost efficient manner.

Facing expanding service requirements of wireless applications and emerging standards, research on wireless telecommunications and radio engineering keeps solving fundamental problems:

(1) How to consume the spectrum more efficiently?

Traditionally, the regulation (or policy) of licensing and utilizing the spectrum leads to a static and inefficient usage. Hardly-predictable processing technology trends and market demand lead to today's unbalanced utilization and shortage of spectrum. It is necessary to introduce an innovative spectrum licensing and coordination infrastructure to enable a flexible, open and dynamic way of utilizing the overall available spectrum resources. Accordingly radios with more functional flexibility and real-time reconfigurability are needed to support such more efficient spectrum behavior.

(2) *How to enable integrated services and harmonize separate networks?*

Rapidly expanding service demands can not be well supported by defining new discrete standards that cause a future network compatibility problem, but rather by integrating and merging fundamental wireless access network infrastructures like cellular, ad-hoc and broadcasting schemes, and optimizing the communication resource coordination for application services. To coordinate wireless network resources in a way that simultaneously integrates services for various applications under different spectrum regulations, a cooperative, flexible, and reliable communication node is needed. To provide ubiquitous connectivity with integrated access and dynamic services, the radio platform for the communication node should be able to support multi-band, multi-mode reconfigurability, and more importantly, be aware of its operational environment and provide service adaptively.

One promising solution to the problems above is the Cognitive Radio (CR); a radio with an intelligence layer of awareness and learning necessary to achieve optimal performance under dynamic and unpredictable situations. Cognitive radio technology is not about the radio itself; it is about reforming the communication behavior of terminals and network segments so that they are able to adapt dynamically, transparently and securely to the most appropriate means of providing wireless services to consumers. For spectrum, the CR can dynamically adapt its behavior to its awareness of the radio environment and spectrum policy. For wireless service, CR, with its learning capability, supports multi-band, mode-mode applications with cognitive service type and quality. Therefore, CR enables an interactive, harmonized way of using the spectrum and communication resources between technology, market, and regulation.

1.2 Cognitive Radio Research History Overview

In such a rapidly growing field, it is difficult to provide a comprehensive history, and authors who write review articles like this run the risk of leaving out important work of which they were unaware. For a fully comprehensive review of the history of CR and current research, the reader should see [2] and [3], both of which provide a good overview of cognitive radio research.

Figure 1.1 provides a brief timeline of cognitive radio research. It also serves as a functional description of the enabling technology transitions that have occurred since Mitola introduced the term "cognitive radio" in 1999 [4]. This timeline includes the major milestones in cognitive radio research activities both globally and from the perspective of our own group. These range from policy / regulation reform to intelligent wireless communications. It also shows increasing density as we try to represent the range of quality work being pursued.

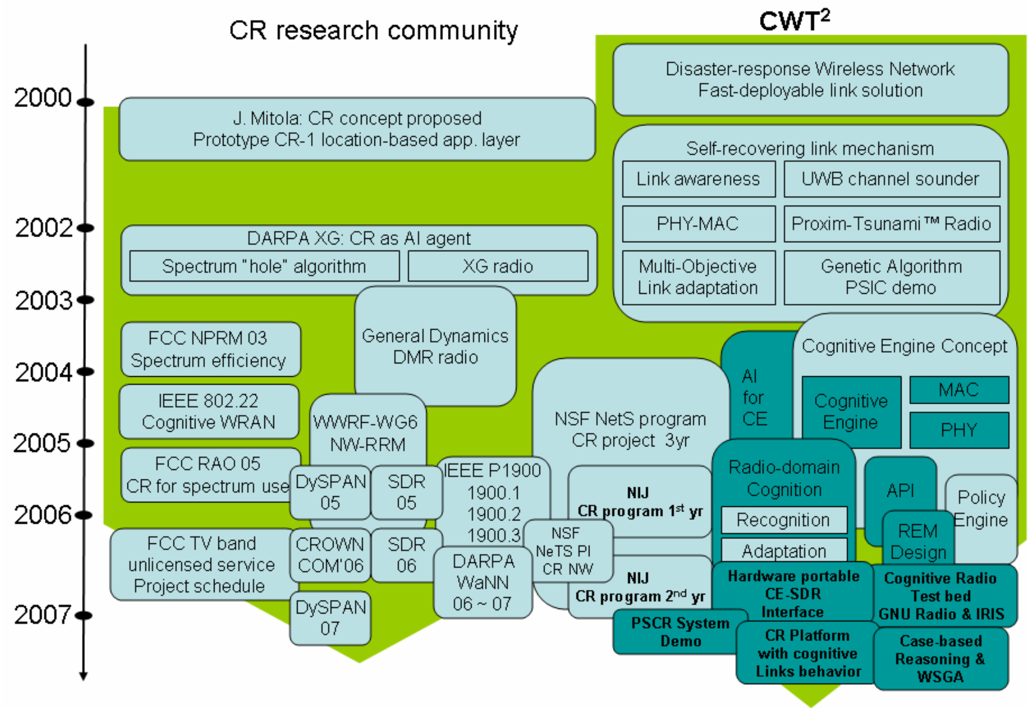


Figure 1.1: CR research history road map

Virginia Tech started research and development of rapidly deployable broadband wireless communications for disaster response in the late 90's [5]. After September 11, 2001, it was evident that there is a need for self-healing wireless networks and radios that could autonomously and legally evolve in time to recover the link. At that time, wireless technologies that allowed radios to adapt their behavior were based on pre-set algorithms, which often failed to function properly or performed poorly when facing unanticipated operation scenarios and radio environments [6]. As research progressed towards highly-flexible radios based on Software Defined Radio (SDR) technology, the idea of creating an intelligent radio that can learn the environment and evolve its capability seemed reasonable [7].

Such *cognitive radios*, a term first coined by Joseph Mitola III [8], have become a topic of great research interest in the past six years. While Mitola envisioned the cognitive radio as a universal and highly intelligent wireless personal digital assistant, operating primarily at the application level, research in the Center for Wireless Telecommunications (CWT) at Virginia Tech started to realize the radio's cognition at the Medium Access Control (MAC) and Physical (PHY) layers [9].

Noticing the unbalanced spectrum usage, in 2002, the Defense Advanced Research Projects Agency (DARPA) funded the NeXt Generation (DARPA-XG) program [10]. The program's purpose was to define a policy-based spectrum management framework inside adaptive radios that can sense and share the use of spectrum, with a focus on policy-based negotiation and radio etiquettes that leverage spectrum "holes" existing in space and time. These XG radios did not have cognitive capabilities as defined above, but could serve as potential hosts for policy-adaptive wireless services. Both the XG program and the real spectrum utilization started drawing the attention of Federal Communications Commission (FCC), whose policy makers sponsored a research that confirmed the underutilization of spectrum in time and space [11]. Later, the Commission filed a Notice of Proposed Rule Making (NPRM) [12], allowing the exploration of cognitive radio technology to facilitate the improved spectrum efficiency and seeking comments on how to manage the rules and enforcement policies for better spectrum utilization. After the NPRM was issued, the FCC filed a Notice and Order (NAO) to facilitate the opportunities for better spectrum use, employing cognitive radio technologies [13]. In 2004, Wireless World Research Forum (WWRF) Working Group 6 started to investigate innovative solutions for spectrum and radio resource management (RRM) by exploring CR technology. Their research focuses on network reconfigurability, and RRM indicates the potential of CR technology for the next generation wireless access infrastructure [1].

The number of conferences and technical sessions on CR has increased rapidly since 2005. The Software Defined Radio Forum's (SDRF) technical conference has offered multiple CR tracks in the past few years. The Institute of Electrical and Electronics Engineers (IEEE) in 2005 introduced the Dynamic Spectrum Access Networks (DySPAN) conference; a meeting dedicated to both technical and regulatory challenges of spectrum sharing and allocation, much of which used or suggested cognitive radio solutions. In 2006, IEEE sponsored the first conference on cognitive radios, CROWNCOM, bringing in new perspectives from a diverse set of researchers from around the world.

Besides active conference participations, IEEE also started standardization efforts of leveraging cognitive radio technology. In 2004, the IEEE 802.22 working group was formed to define the Cognitive Wireless Regional Area Network (WRAN) MAC and PHY specifications [14]. At the end of 2005, the IEEE launched the Project 1900 standard task group for next generation radio and spectrum management [15], with a special focus on applying software defined and cognitive radio technology. As IEEE started 802.22 with a special interest of defining procedures for cognitive operation in TV bands; after three years of preparation, FCC launched the TV band unlicensed service project in 2006 with cognitive radio technology [16].

In the summer of 2004, the National Science Foundation (NSF) initialized a program for research in Networking Technology and Systems (NetS) programmable wireless networking [17], which points to a promising future of cognitive radio research. At the same time, National Institute of Justice (NIJ) launched the Communication Technology (CommTech) program on investigating wireless global, flexible and secured interoperability solutions for national police and fire departments [18].

In 2004 CWT demonstrated a biologically inspired Cognitive Engine (CE) based on the Genetic Algorithm (GA) that is capable of intelligently adapting a radio's PHY and MAC behavior in the face of unanticipated wireless and network situations [5, 19]. From 2004, I joined CWT's CR research. Specifically with my research focus on defining a general cognitive radio node solution for various cognitive wireless applications, I started the CWT Cognitive Wireless Technology (CWT^2) research thread in spring 2005. Since summer 2005, CWT^2 has been funded by NIJ CommTech and NSF NetS programs, and later expanded to include a research team led by me from August 2006. Since that time, with joint work with T. W. Rondeau and C. W. Bostian, the Cognitive Engine concept has been evolving as a general software solution for the cognitive radio node development. A generally applicable machine learning core in radio domain has become a key part of my research focus [20]. From 2005 to 2006, I conducted interdisciplinary research as the theoretical preparation for defining and designing a cognitive radio node. The research topics mainly include artificial intelligence techniques like case-based reasoning, artificial neural networks and evolutionary computation, stochastic signal analysis in time and frequency domain, software defined radio engineering issues like Analog-to-Digital Converter (ADC), multi-rate processing and diversity processing. Aside from system development efforts, the research contributed publications and tutorials in each area for the cognitive radio research community. In the SDRF conference of 2006, CWT^2 demonstrated a prototype smart receiver including a General Purpose Processor (GPP) based software defined radio platform, signal classification capability and PHY-MAC reconfigurability with hardware-independent radio interface [21]. In April 2007, CWT^2 demonstrated a fully-functional cognitive radio node prototype for public safety applications at the NIJ CommTech Program Review in Las Vegas. With the success of node design, CWT^2 is extending the current cognitive engine system from node centric to network oriented processing structure to enhance the support for cognitive network capability [22].

Recent CR research advances also include the Implementing Radio in Software (IRIS) platform from the Center for Telecommunications Value chain Research (CTVR) of Trinity College, Dublin, Ireland [23, 24]. Ireland has made a significant contribution to both the SDR and CR research communities by opening two 25 MHz slices of spectrum for research in spectrum utilization. CWT and CTVR are collaborating on a series of projects to enable interoperability between SDRs and CRs, which leads cognitive radio research towards cognitive networks [23, 25] [22].

1.3 Some CR Research Challenges

The recent progress in wireless technologies indicates some enabling power for cognitive wireless applications. From the wireless communications aspect, both classical theories and modern advanced signal analysis methods pave the road toward the development of waveforms, signaling techniques, and network topologies to meet various service needs. From a radio engineering aspect, both analog and digital processing techniques are advancing along with supporting semiconductor technologies, improving the performance-cost ratios of different system implementations. However, there are challenges in applying currently available technologies for cognitive radio development. Cognitive radio technology features coverage in both depth and width: it targets both user service satisfaction and radio platform optimal design; it incorporates a certain level of artificial intelligence by computational technology with wireless communication principles subject to engineering implementation limitations. Some of the research challenges are briefly addressed below.

1.3.1 How Much Intelligence is Enough for The CR?

Defining “cognitive radio” is as difficult as defining “artificial intelligence.” Intuitively, the CR should exhibit intelligent behaviors. However, “cognitive radio” is still a vague term in the CR research community. There are different CR definitions with different emphasis on radio behavior, operational objective, or the scope of the target problem. For example, Joseph Mitola, who coined the term “cognitive radio”, envisions the cognition level in which wireless terminals and related networks are sufficiently computationally intelligent about radio resources and related computer-to-computer communications that : (1) they detect user communication needs as a function of use context, and (2) they provide radio resources and wireless services most appropriate to those needs [8]. Such a definition describes how intelligent a CR should be, while not providing much insight into the required functionalities. Simon Haykin defines a cognitive radio by a long description of its behavior and its performance objective, but without a clear explanation of reasoning and learning [3]. The FCC simply suggests that any radio having the adaptive spectrum awareness should be referred to as a “cognitive radio.” There are other CR definitions from DARPA, SDRF, and other organizations.

Although one of the major motivations of CR research stems from the demand for efficient spectrum management— given that spectrum licensing / allocation policy is now severely unbalanced and obsolete—that the FCC is most interested in, CR is much more than being spectrum-agile. The power of cognition turns a conventional radio into an intelligent agent that can learn its environment and adapt itself accordingly in all controllable operation parameters to optimize its performance. And spectrum agility is only one dimension of its optimization domain.

Despite definition variations, it is generally agreed that a cognitive radio exhibits the intelligent capabilities of being aware of its environment, adapting its operation and learning from its practice feedback [26]. However, these terms are not precise enough to address a system design requirement. For example, “aware” may include awareness of the RF environment, awareness of its own capabilities, awareness of the rules that govern its operation, awareness of its user’s priorities and authorities, etc. “Adaptive” may range from simple transmitter power control to selecting from a menu of standard waveforms to creating new waveforms and protocols on the fly and negotiating their use with another radio like itself. “Learning” may range from experience-weighted table lookup to arbitrary combinations of machine learning algorithms. Determining what intelligence level is needed is a fundamental goal of CR system design.

CWT^2 CR system design follows three basic rules:

(1) The theoretical investigation is scoped by the canonical wireless communication problems, including spectrum-oriented tasks like finding and using temporarily vacant spectrum ("white space"), and service-oriented tasks like providing near-universal interoperability by identifying and interoperating with legacy waveforms and adjusting waveform parameters to optimize figures of merit like throughput, bit error rate, power consumption, spectral footprint, etc. Thus the cognitive capability is not designed to prove the completeness of machine learning philosophy, but to optimize radio resource utilization and communication service quality.

(2) System design and implementation are tailored to fit the specific target application scope. In other words, the intelligence level is designed to be useful enough for the service need. Any extraneous degree of intelligence the radio possesses may cause unwanted cost in computation and time. For example, CWT^2 designed a complete cognitive radio software testbed architecture, but only the essential functional modules are implemented to form an efficient system for a specific target problem, like the Public Safety Cognitive Radio (PSCR) node [27].

(3) In its operation, CR only becomes cognitive when needed. During standard operations which do not require intelligent behavior, the CR simply acts as a standard radio to avoid an unnecessary efficiency penalty. Such flexibility inside machine learning is realized in the CWT^2 CR using a learning core combined with both pattern match and evolutionary search methods for solution making. Under different problem scenarios, the learning core can adapt its learning behavior with these two methods for best performance trade-off.

1.3.2 Bridge AI with Radio Engineering and Wireless Communication

After experiencing emerging problems in spectrum utilization, interoperability between different standards, and poor resource management in wireless networks, the computer science world [4, 8] is proposing a radio that assumes all necessary sensing capabilities for the awareness of comprehensive environment information. It also assumes that the radio has enough intelligence to reason and learn all collected information, and make considerate decisions that can take care of almost everything for the user. Such a vision is too profound to be practically supported by radio technology in the near future. Meanwhile, the radio engineering world (both academic and industry) is still at the stage of pushing radio's functionality from hardware to software implementation in order to increase the flexibility for various communication scenarios [28]. Such a software-defined solution confines itself within the scope of a limited set of pre-programmed adaptations. When the radio is deployed in an unknown situation, it cannot adapt.

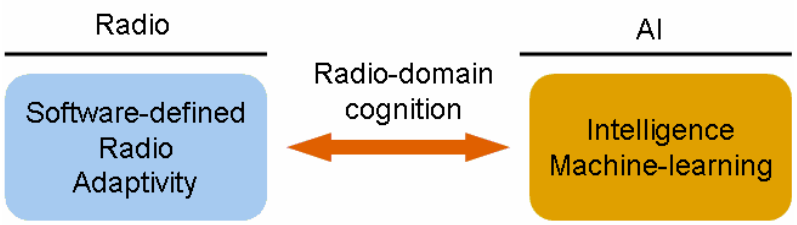


Figure 1.2: The bridge of radio-domain cognition between AI and radio platform

We need a cognitive radio that not only performs mechanical (i.e., by rote) adaptation, but also learns its environment and spontaneously adapts itself to changing or unknown conditions. Therefore, a machine learning a layer consisting of awareness, reasoning, and adaptation is needed in the radio domain, serving as the bridge between theoretical Artificial Intelligence (AI) and a practically limited hardware platform. Such a cognition layer should understand the radio's resource availability, reconfiguration, and control to optimize its performance for required waveform generation. This imposes challenges in both radio platform modeling and intelligent algorithm design, especially for heterogeneous radio hardware and cross-layer optimization objectives. A general method is needed so that various radio hardware platforms can be modeled in the same way so that general machine learning mechanism and knowledge structure can be applied. To jointly optimize performance across multiple communication layers, the layers' implementation in the specific radio platform should be modeled and evaluated to support general learning.

In the CWT^2 cognitive radio solution, a software defined approach is chosen for the radio platform design for maximal flexibility. Software defined implementation of waveform generation and communication stack control enables flexible, on-line reconfigurability. A platform independent radio interface is developed between radio and intelligent algorithms, so that general machine learning methods can be applied without the limitation of specific radio implementation.

1.3.3 Pushing Theoretical Research Toward the Engineering World

It is usually more difficult to prove that a new idea is practically suitable for engineering than brainstorming it. For an architectural innovation like cognitive radio technology, it is important to make sure that its theoretical framework is generally applicable so that resulting engineering solutions can be adaptively designed to suit different target problems, as far as the problem is within the overall application domain. Also, an engineering solution development lifetime should be from theoretical innovation to practical system implementation and functionality testing. Pushing theoretical innovation to the practical running of a prototype involves a great amount of engineering effort; it is also the best way of verifying the theory.

Through three years (2004 to 2007) of work, CWT^2 developed a complete cognitive radio node solution, which features a general radio domain machine learning core, sensors that are aware of radio environment, user needs and radio platform, and a general radio interface that can apply cognitive algorithms on any reconfigurable radio platform. Also, during the development of the cognitive system, CWT^2 developed a PHY-MAC fully reconfigurable software defined radio system as the supporting radio platform. This CR system is to serve as the node solution for different cognitive wireless networks such as a public safety cognitive network, NSF NetS project for cross-layer performance optimization, and DARPA Wireless Adaptive Network Development (WAND) project.

This research illustrates a development thread from architecture definition to prototype implementation, which can serve as a preparation and reference in both theoretical exploration and engineering experience for the CR research community.

1.3.4 Interdisciplinary Background Needed

An additional challenge with cognitive radio research is its interdisciplinary nature. A relatively large coverage of theoretical background is required. Taking CWT^2 CR research as an example; a basic list of needed background knowledge is provided below.

(1) Artificial intelligence
Case Based Reasoning (CBR) theory is applied for radio domain solution making; Artificial Neural Network (ANN) techniques are applied for radio environment recognition; Genetic Algorithm (GA) techniques are applied for multi-objective solution search under strict radio domain conditions; a relational meta-database is implemented to construct the cognitive radio's knowledge base; object oriented modeling and language are used as the general interface between cognitive functional elements; web based information processing is applied to support distributed intelligence.

(2) Wireless communication theory and signal processing
For radio environment awareness, stochastic process analysis and linear algebra processing methods are needed to model communication signals, and digital complex signal processing in both the time and frequency domains is needed for signal analysis. For adaptive signal processing, multi-rate digital signal processing is needed for multi-mode waveform generations; adaptive filtering and equalization algorithms are needed for propagation channel modeling and compensation. Space-time-frequency diversity

processing may also be needed if Multiple Input Multiple Output (MIMO) and multi-carrier communication techniques are considered for radio platform implementation.

(3) Software defined radio engineering

Object oriented software engineering for radio system implementation; general Application Programming Interface (API) and event handling design; multi-threading design to realize real-time reconfigurability and control hierarchy. Software validation and functional verification is important for the legality and reliability assurance of software defined waveform generation. Computation optimization and cost profiling is also important for embedded implementation.

1.4 Personal CR Research Experience and Contribution Overview

I joined CWT at Virginia Tech in 2003 as a Master of Science graduate student with a research focus on direct conversion receivers. This led me to a deep understanding of wireless transceiver infrastructure and design challenges [13]. This experience later helped me identify the key problems in waveform recognition design for cognitive radios. After writing a white paper on receiver structure [14], I joined cognitive radio research focusing on how to realize the cognitive capabilities at PHY and MAC layers. I began to discover how to realize the widely said “cognitive capabilities” on a practical radio platform. How could we make the radio learn, i.e., to sense, reason and make decisions to control the radio adaptively, and improve its knowledge through learning? At that time, most literature of cognitive radio technology either complained about an existing spectrum problem, or plotted a beautiful picture of next generation wireless networks by using imagined cognitive capabilities. The question I asked myself was “forget about the future goals, let’s take the first step – how can the cognitive functionality using currently available communication theory and radio technology be realized?” Therefore, since the beginning of my research in cognitive radio, I have pursued a practical engineering solution for realizing radio cognition in the PHY and MAC layers.

Starting from the learning core of cognition, I felt, in 2004, that the well-cited Mitola's cognitive cycle is difficult to be directly applied for PHY and MAC layer learning and adaptation, as well as for today's embedded system implementation. Therefore, I designed a cognition cycle with simple topology but complete learning capability, which is directly targeted for on-line processing on the SDR platform. The cognition cycle has an inner loop of general machine learning and an outer loop of radio domain recognition and adaptation.

The environment that the radio needs to learn is complicated, including radio environment, user requirements, spectrum policy, and even the radio itself. Such a complicated environment makes it very difficult to design general machine reasoning and learning algorithms to take care of all possible scenarios; additionally, the learning system has to be very complicated and requires huge computational cost. To ease the radio learning problem, I segmented the overall environment into three domains: radio domain, policy domain and user domain. And I have been focusing on the cognition in radio domain ever since.

In order to bridge the machine learning theory with radio engineering technology, I designed, in 2005, a two-layer radio domain intelligence system to realize the cognition [20], which is an exact implementation of the two-loop cognition cycle. Such a hierarchical system separates the radio-domain's specific operations, such as wireless signal detection and radio link control, from the general machine learning core. The software system implementing the cognition was termed "Cognitive Engine (CE)," a term and concept created by Christian Rieser, a former CWT graduate student. [7].

Since the summer of 2005, CWT's CR research has been funded by the NIJ CommTech program and the NSF NetS program. Specifically, my CR research, primarily under NIJ CommTech project grant, has two-fold responsibilities: (1) to provide Public Safety Cognitive Radio (PSCR) solution to interoperability and intelligent radio service for the emergency response and public safety community, and (2) to provide a general cognitive radio node solution to cross-layer optimization for various adaptive and cognitive network applications.

Radio environment recognition is a key component in the outer layer of the designed hierarchical cognition system, providing environment information to the inner layer learning core. The CR needs to recognize an incoming signal without enough prior knowledge, I designed a two-stage waveform recognition sub-system [21, 29]. I also looked into adaptive and blind algorithms for channel equalization. Based on this sub-system, a smart receiver prototype was successfully demonstrated at the Software Defined Radio Forum (SDRF) Technical Conference in November, 2006. This sub-system is now already implemented in full C++ and integrated in the PSCR node prototype.

The cognition system learning core consists of reasoning and solution making. It is my understanding that to achieve a usefully enough intelligence level, the cognition in radio domain is within the scope of a goal-driven resource-limited machine learning system [30]. It's important to realize that the radio domain not only features limited resources and thus bounded operational space, but also involves hybrid, complicated domain knowledge. It could be difficult to elicit generally applicable rules like those used in a conventional expert system. Therefore, I decided to develop a Case Based Reasoning (CBR) system with declarative knowledge representation (i.e., the case database), which gives more flexibility in both knowledge encoding and, more importantly, data mining and decision making by a "pattern matching" approach. To enhance the solution making to respond to novel or new situations, I applied the Genetic Algorithm (GA) to aid the solution adaptability. GA's evolutionary searching strategy is suitable for a not-well-known problem space with complicated domain rules due to its powerful problem encoding.

In implementing the CBR-GA combined solution chain, I used standard the Sequential Query Language (SQL) database to support distributed processing for future network oriented cognitive behavior; and work with another CWT student, T. W. Rondeau, on

Multi-Objective Genetic Algorithm (MOGA) development, which resulted in an outstanding paper award at the 2004 SDRF conference [31]. After that, I continued designing an adaptation mechanism to speed up GA search and applied it for a time-strict optimization problem, the dynamic spectrum access optimization, with a significant performance improvement [32].

In designing the Cognitive Radio Knowledge Base (CRKB) for reasoning and learning, I pointed the design goal toward the cognitive radio network intelligence. While working with Y. Zhao from the Mobile Portable Radio Research Group (MPRG) at Virginia Tech, we designed a hyper-dimensional database system, Radio Environment Map (REM), to serve as centralized or distributed network intelligence for the resource management and traffic coordination of the cognitive wireless network. This work became Chapter 11 in [26]. The REM is implemented as one part of CWT^2 CRKB, it is also implemented and extended by Zhao as a pure network knowledge base [33].

During the development of radio domain cognition system, I realized I needed a highly flexible radio platform for functional support and verification. I picked the SDR platform due to its software-based flexibility, and started development based on open source GNU Radio software package [34] due to its open source nature and GPP-Linux environment. Using GNU Radio signal processing, C++ library, and Python threading flow graph, I designed a PHY-MAC reconfigurable software defined radio system called CWT^2 waveform framework. It serves as the general testbed for algorithm development and prototype demonstrations. Now it is integrated into the PSCR node prototype release. In preparation for the radio development, I conducted a system level design trade-space investigation of SDR implementation, which resulted in a 2006 full-year No.1 monthly downloaded paper in IEEE [35]. This research also led to a joint proposal recently funded by DARPA [36].

Several other students joined CWT^2 at the prototype integration stage since October 2006. Leading the CWT^2 team working on multiple implementation threads of constructing the complete cognitive radio node system gave me a great challenge, but also gave me precious management experience, especially the multi-tasking coordination and quality control from system to block level. The full-functional PSCR node was demonstrated successfully on April 24, 2007. And ten days later, on May 4, 2007, CWT^2 published its first release of PSCR node solution software package (version 1.0).

Looking back to the past three years, my research has focused on providing a general cognitive radio node solution to realizing intelligent wireless communication, in which the cognitive radio operations enable a flexible, open, and secured way of utilizing the spectrum, and the cognitive wireless network provides fully-integrated service for the user anytime anywhere. Through the road from architecture definition to final prototype integration, all my efforts are not spent on proposing fancy ideas, but on preventing mistakes, and on providing engineering solutions that are reasonable in theory and feasible in implementation. A comprehensive list of my research contributions is provided in Chapter 8.

1.5 Organization

This dissertation is organized into eight chapters and two appendices. Chapter 1 first introduces the current existing problems of the wireless communications world and the motivations of CR research. It then provides a CR research history road map from its origin to today. Several fundamental research challenges are discussed in Chapter 1. It ends with a brief overview of the author’s CR research experience and contributions.

Chapter 2 begins with a general CR solution with the definition of an innovative CR architecture. Its hierarchical intelligence structure and the resulting advantages are explained. Then Chapter 2 gives an introduction of CR related domain knowledge background with a focus on the radio domain. A set of radio domain cognition related radio functionalities is defined. Then the cognitive algorithm software system, the cognitive engine (CE), is defined, and its system structure is explained in detail. Further inside the CE the learning core of cognition is defined and explained.

Chapter 3 focuses on how to achieve the awareness in the radio environment. A set of radio environment recognition functionalities are defined. The radio environment recognition mainly includes the spectrum scan and waveform format identification at the channels of interest. For the spectrum, an adaptive block FFT based frequency sweeper and energy detector is introduced; for the waveform, a two-stage adaptive signal classification system is introduced. After giving the details of these two functional modules, two resulting simulation and prototype testbeds are described with their demonstration setup and scenarios.

Chapter 4 addresses the design of machine learning for the CR. It first introduces fundamental machine learning principles and general AI algorithms suitable for radio domain cognition. Then it describes the designed reasoning and learning structure which combines case based reasoning (CBR) and genetic algorithms (GA). The designs of CBR and GA for machine learning are detailed one after another. Then the CBR-GA chain for CR learning core integration is provided. To support the machine reasoning and learning, radio domain knowledge is defined and formulated at the end of the chapter.

Chapter 5 introduces a platform independent interface for the cognitive engine to work with different radio platforms. The importance and advantages of having such an interface are explained with the emphasis of knowing and controlling the radio platform system. Then the design and implementation of this interface is given.

Chapter 6 gives a complete system design of a reconfigurable SDR platform, called CWT^2 waveform framework using GNU Radio library and USRP radio front-end. It first discusses some general SDR design trade-offs that are closely related to the target radio platform to be developed. Then it provides a brief background introduction of GNU Radio and USRP, with focuses on why and how they are used in designing the waveform framework. In the center of the chapter is a comprehensive development thread from global system level to element signal processing algorithm level. Hierarchies of configuration, control and signal processing are detailed, and key design trade-offs are clarified.

Chapter 7 describes the overall CR node system integration with all the building blocks available from previous chapters. A complete CR node system is constructed and shown in detailed block diagrams. Following this architecture, an application specific CR system, the Public Safety Cognitive Radio (PSCR) node is designed for public safety communication interoperability. The complete PSCR node software system is packaged to an official release including installation guide and user/developer manuals. The implementation of the fully-functional PSCR node prototype is also explained. This chapter ends with a description of the PSCR node functionality demonstration in the recent NIJ CommTech Program Review.

Chapter 8 first summarizes the author's CR research contributions and related publications. Then it points to several future research objectives from current achievements.

.

There are two appendices attached to this dissertation. Appendix A provides a list of acronyms, and Appendix B provides the code details and related information for the PSCR node system.

Chapter 2: Cognitive Radio System Architecture

2.1 CWT^2 Cognitive Radio Architecture

2.1.1 CWT^2 Cognitive Radio System Solution Overview

The *Oxford English Dictionary* definition of "cognitive" is "pertaining to cognition, or to the action or process of knowing," and "cognition" is defined as "the action or faculty of knowing taken in its widest sense, including sensation, perception, conception, etc., as distinguished from feeling and volition". Therefore, the process of sensing and understanding an existing wireless environment, then deciding a radio's operation to adapt to the perceived world and evaluating the result of the practice is a cognitive process. Based on such understanding, we define a cognitive radio *as an intelligent communication device that is aware of its environment and application needs, and can reconfigure itself to optimize quality of service* [20].

Following the cognitive radio definition, a simple question-and-answer list is provided to give an overview of CWT^2 cognitive radio system solution.

(1) *How to make a radio cognitive?*
We designed a cognitive radio software package, called a Cognitive Engine (CE), overlaid on radio hardware platform. CE manages radio resource to accomplish cognitive functionalities and adapts radio operation for performance optimization.

(2) *What is the cognition model?*
A specific machine learning model (embedding the two-loop cognition cycle as the learning core) is designed to enable cognition capability for wireless applications. Case based reasoning and evolutionary search are combined in the learning process.

(3) *Which host radio architecture?*

Any software defined radio platform and any radio with a certain level of reconfigurability can be supported by a cognitive engine with the platform independent radio interface.

(4) *Which communication layers have cognition?*

Currently the cognitive radio functionality is focusing on the PHY and MAC layers for cross-layer optimization. The designed cognition algorithms can easily be extended to the network and application layers due to its general learning core.

(5) *How to establish a cognitive wireless network?*

CWT^2 provides a cognitive radio node which can be deployed for both centralized and distributed network intelligence. As network nodes, cognitive radios can work individually or jointly on resource management and performance optimization.

2.1.2 Radio Platform Independent Design Approach

In previous work by CWT in 2004, C. J. Rieser, T. W. Rondeau, C. W. Bostian, and others proposed a bio-inspired radio cognition model by applying evolutionary computation techniques for smart radio link adaptations [7] [9]. In 2004, I joined CWT's CR research and started working with T. W. Rondeau. While Rondeau continued focusing on applying the genetic algorithm for radio domain performance optimization, I initialized the CWT^2 research thread of developing a general machine learning system that can be applied for different cognitive radio applications. A general cognitive radio solution is defined in the form of a software package that can work with general reconfigurable radio platforms to provide cognitive functionality. This cognition software system is named the Cognitive Engine (CE).

Figure 2.1 shows two views of CWT^2 cognitive radio system structure. The block diagram view on the left explains the relation between the CE cognition system and the radio platform, and the behavioral diagram on the right illustrates the cognitive functionality of combining the machine learning process with radio operation. The radio's receiver senses the radio environment and reports to the CE using a standard parametric format called "meters." The CE makes a decision according to these meters, and passes the decision to the radio with another standard parametric format, called "knobs". Then the radio carries out reconfiguration to meet these knobs' values. This scenario can be simply expressed as "The CE reads the meters from the radio and turns the knobs of the radio."

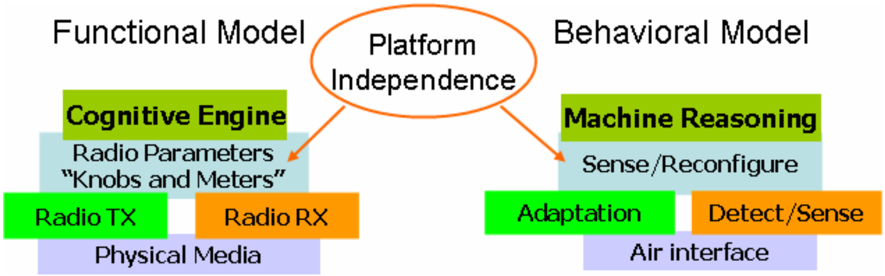


Figure 2.1: Structural and behavioral views of the CWT^2 cognitive radio system

This meter-knob concept illustrates a hardware independent interface that has been designed to bridge arbitrary cognitive engines and radio platforms. It has two major advantages: it supports general machine learning and application specific algorithms, and it supports heterogeneous SDR or reconfigurable radio platforms.

This hardware independent interface consists of general radio configuration and control APIs, standard waveform and platform representations, and supporting information parsing modules. It serves as the bridge between artificial intelligence and radio engineering and largely defines the CR overall system hierarchy. Such a structuring is illustrated in an egg model in Figure 2.2.

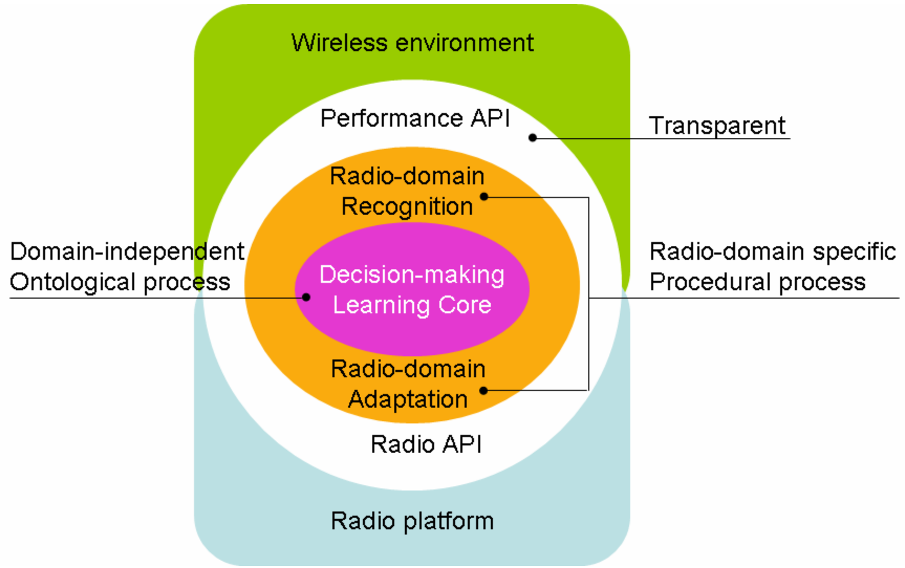


Figure 2.2: Platform independent CR system hierarchy: the Egg Model

At the AI side, algorithms need not to be platform specific; thus general knowledge and learning can be freely applied for a variety of applications' problems. At the radio side, different radio platforms can be enabled to support cognitive functionality so far as they provide the needed reconfigurability. In the CWT^2 CR node, this interface is implemented as a general radio interface supported by a specific set of radio domain knowledge and detailed in Chapters 4 and 5.

2.1.3 Network Oriented Functional Structure

Rather than considering the cognitive radio as a sole entity, we should think of it as part of a network. A CR is a node along with other network nodes [37], including other CR nodes as well as nodes without intelligence. In such a network cooperative reasoning, learning, and adaptation with information sharing can be conducted across multiple nodes in the network. Thus the cognitive capability extends up into the network, transport, and application layers in addition to MAC and PHY layers.

Network support is important to the evolution of wireless communications from legacy devices to cognitive units, and from the incompatible radio networks to the integrated cross-network services. The cognitive capabilities should be general and flexible enough to support networks with different structures and service requirements. Either a legacy system that has limited or even no intelligence or a CR system that is short of resources like memory and computational power can simply request information from CRs on the network. Such information can be a fact map to fill a local knowledge hole for a CR, an adaptation strategy for a reconfigurable but not cognitive radio, or simply an on/off link control of a legacy radio.

As explained in the rest of this chapter, the CWT^2 CR learning structure consists of an information processing chain with three steps: recognition, reasoning and adaptation. These steps can be flexibly implemented in either a centralized way as a fully functional CR node or be distributed across the network where different local parts of the network require different levels of intelligence and different layers of optimization. Such a radio cognition functional structure is illustrated in Figure 2.3. For instance, one radio may contain global environment knowledge while another one offers better computational power for reasoning and solution making. Such cooperative information processing reduces the complexity any single node requires while still benefiting the entire network with distributed cognitive functionality. In this way, we are making a cognitive network.

In the CWT^2 CR node structure, not only the information processing chain can be distributed but the processing modules themselves can also be implemented across the network. For recognition, a distributed sensing approach results in cooperative radio environment awareness. For reasoning, a network based knowledge implementation provides more comprehensive and reliable information for cross-layer performance optimizations, which always involve a group of nodes' interactive behavior; and its distributed database implementation can facilitate information sharing and joint adaptation. It also helps prevent node based problems like the single-point failure and the hidden node problem. To generate adaptive optimal solutions, evolutionary search is applied. It is well known to have the parallel processing nature that leads to a multi-objective optimum Pareto front in convergence as well as robust global search against local traps [38].

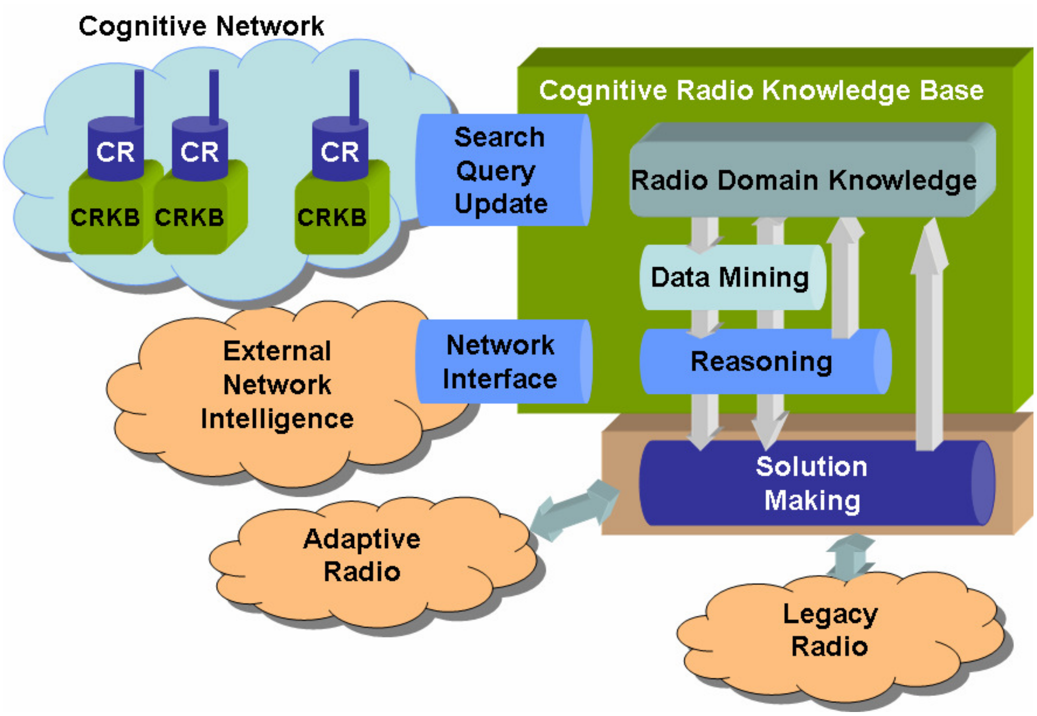


Figure 2.3: CR node functional structure for distributed network intelligence

2.2 Domain Knowledge Partitioning

Cognitive radio design is highly multi-disciplinary research area, combining artificial intelligence, wireless communications, computer science, spectrum regulations, and service marketing, and many others. In a narrow sense of the engineering knowledge background, CR technology mainly involves three fields, the policy domain, the radio domain, and the user domain. The CR needs to be aware of the radio environment and operation policy and automatically optimize the performance by understanding the user's needs. As shown in Figure 2.4, the CR sits at the center and integrates the knowledge from three domains.

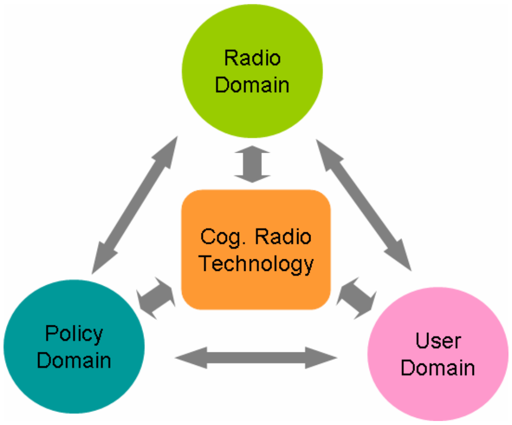


Figure 2.4: Cognitive radio technology related domain knowledge

The policy domain contains information regarding the policies that govern the radio's operation. Such information may be predefined and stored in the CE's knowledge database or it may be downloaded in the field. It is used to guarantee the security and legality of the cognitive radio's operations.

Regulatory issues of radio operation, like the frequency plan, transmit power and interference limits, are taken into account as a hierarchical-structured (tiered by priority) rule set used as constraints in the radio operational space. Such a rule set can be extended to the MAC and network layers to optimize network performance.

The user domain defines both the operation preferences and performance requirements from both the service provider and end user. It mainly provides performance objectives like access availability, service type and Quality of Service (QoS). The CR needs to understand these requirements and try to meet them by adapting its operation.

In the radio domain, there are mainly two types of information that the CR needs to be aware of: the radio environment and the radio platform. The corresponding external and self awareness are combined to feed machine reasoning in two forms: either to provide objectives if some radio function features can be utilized under certain observed environment conditions, or else constraints if radio platform resource limitation becomes an issue. For example, the transmitting power will be minimized if the CE knows that the radio is battery-driven although the user does not specifically place power requirements.

It is useful to partition this complicated knowledge background into several domains according to their emphases and related cognitive operation. By partitioning the knowledge, cognitive functionalities can be modularized and the complete intelligence system can be designed with an open structure. An open structure is important for a machine learning system to modularize specific algorithms for specific target problem, while still maintain a general framework. The CWT^2 Cognitive Engine software system exactly reflects this design approach, explained in the next section.

2.3 Cognitive Engine Software System Overview

In the CWT^2 CR solution, the CE software system is shown in Figure 2.5.

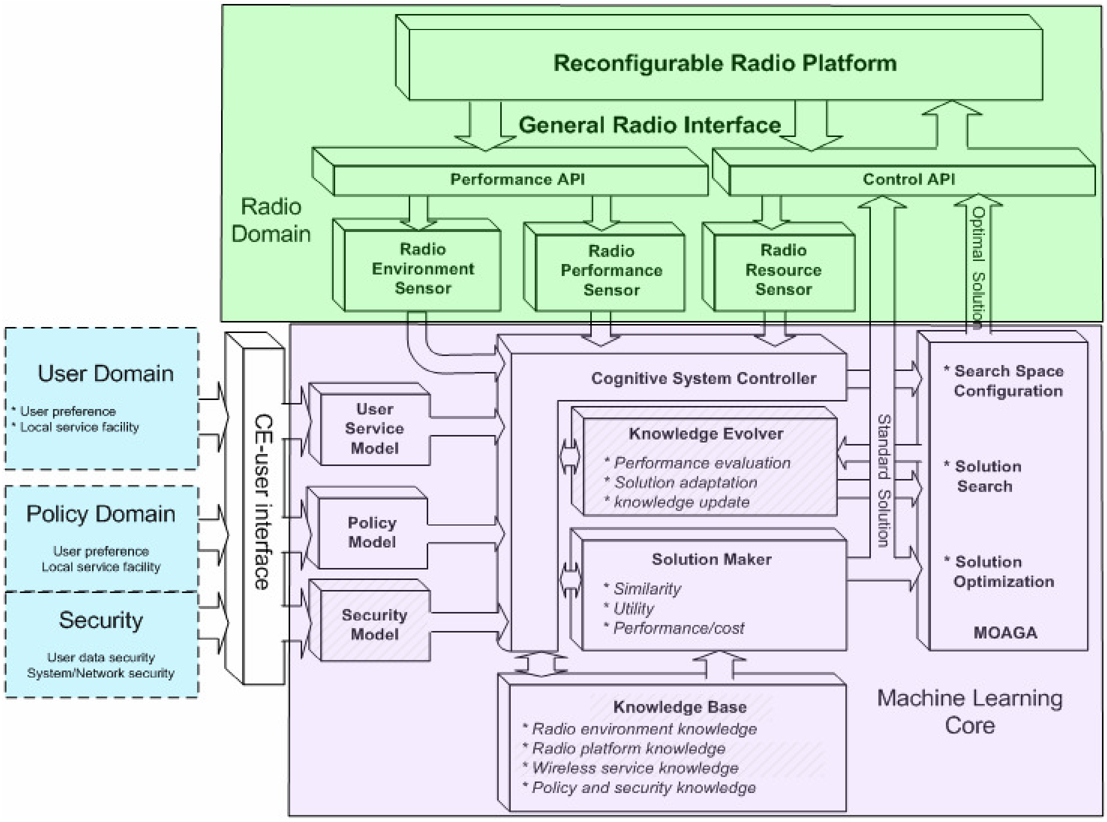


Figure 2.5: Cognitive engine system block diagram

The CE system follows a modular design. There are mainly three sets of functional modules in the CE:

(1) Environment modeling modules handle information collection and modeling for specific domains. As the whole problem space is partitioned into radio, user and policy (including security) domains, the CE models each domain and reports combined environmental awareness to the learning core. For the radio domain, the CE provides a set of sensors which collect radio environment information like waveform features, interference and propagation channel conditions, transceiver performance indicators like error rate and throughput, and radio platform resource status indicators like power consumption and computational cost (see Chapters 3 and 4.). For the user domain, user service preferences like service type and quality are collected and tracked to form a parametric user service model. Similar models are formed for the policy domain. Such modeled information is important for CE to define the dimensions, range, and boundaries of the solution space.

(2) The learning core consists of a set of modules that execute machine learning operations including reasoning, solution making and adaptation, and knowledge evolving. The solution maker is the kernel module that generates a viable solution according to the current input problem scenario (including the environment, user objectives, available resources, etc.) through knowledge based reasoning. The knowledge base is a database containing domain related data like situational information and performance criteria and general principles of reasoning and learning. The Multi-objective Adaptive Genetic Algorithm (MOAGA) is an evolutionary search module that works with the solution maker for further solution adaptation for performance-critical or novel situations. The knowledge evolver updates the knowledge database with the experience from the new solution. Following the reinforcement learning principle, the CE educates itself by evolving its knowledge from practice, which is the true power of cognition (See Chapter 4.).

(3) The CE provides an interface between itself and each domain. For the radio domain, a platform independent radio interface is constructed with which the CE can monitor, configure and control heterogeneous radio hardware without changing its own algorithms (See Chapter 5.). Both user and policy domains are connected to the CE via a user interface due to the similarity between data collection and modeling. All of these interfaces support both node-centric and network distributed implementations. For example, the CWT^2 PSCR system, although a node solution, provides network based radio and user interfaces that support fully distributed cognitive operation (See Chapter 7).

In constructing the CE, the realization of the functional modules described above relies on the effectiveness of applying a wide set of theories and algorithms that are individually well known but rarely found together: statistical signal processing, artificial neural networks, case based reasoning, multi-objective genetic algorithms, and web based information processing. All of them will be explained in the following text of this dissertation.

2.4 The Learning Core of Cognition in the Radio Domain

2.4.1 The Definition of Cognition in CR Research

Before we go into the details of the CE's machine learning core, it is useful to define some fundamental terms that will be often used as the baseline of the CR research focuses and development emphases in this dissertation.

(1) Cognition:
In the CWT^2 CR system, cognition is defined as a combination of intelligent capabilities including recognition, learning, and optimization. These operations are connected to form a machine learning loop.

(2) Recognition:

Recognition capability is achieved by two procedural operations, sensing and classification. The target of sensing includes both the external environment encompassing things like the air interface, network, user and policy, and the internal environment including things like the radio system and CE self-configuration. Sensed information is modeled into an appropriate feature format and categorized by classification into a finite set of scenarios for machine reasoning.

(3) Learning:

Learning is defined as a capability achieved through a loop behavior consisting of making decisions through reasoning and getting educated from practice feedback. A fundamental prerequisite for machine learning is a method for recording whatever has been learned – the memory of knowledge. A relational knowledge database is preferred to follow the associative nature of memory.

(4) Optimization

Optimization is defined as a capability achieved through refining the solution and adapting practice accordingly. Optimal performance is a goal of such operations by directing the practice toward better results.

2.4.2 The Soul of Cognition: Cognition Cycle

The design objective of machine-learning capability leads to the development of a cognition cycle where artificial intelligence keeps evolving through the loop of reasoning, decision making, adaptation and knowledge accumulation.

As the creator of the cognitive radio term, Mitola also proposed a cognition cycle which is well known and widely cited [8]. However, there are several reasons that the CWT^2 defined a new cognitive cycle instead of using Mitola's. First, Mitola's cognitive cycle is defined at the application layer, with a simplified model of low-level communication layer (PHY and MAC) behaviors and optimistic capabilities assumptions, especially for wireless links and networks. This makes it relatively difficult to realizing its envisioned cognition directly in the radio domain. Second, Mitola's cognitive cycle is defined to realize human-like cognitive capabilities which largely rely on today's super computation. These are well beyond the capability and scope of current radio engineering and device technologies. Third, the functional diagram is a bit complicated for embedded system design as the widely-spread relations are not suitable for efficient finite-state-machine design, and the definitions of functional nodes are too general or vague to be implemented directly with computational techniques.

In 2004, I felt that a straightforward machine learning cycle is needed for radio's cognition, which should be straightforward with clear functional definitions but complete enough to carry general machine learning capabilities for various wireless applications. The skeleton of its autonomous running mechanism is a finite state machine that seamlessly connects the radio platform and artificial intelligence together. Therefore a new cognition cycle was defined [20]. It has two feedback loops that realize two levels of intelligence; it also separates the general machine-learning core from radio-specific operations, which reflects the platform independence of the cognitive engine system, as stated in previous sections.

As shown in Figure 2.6, the cognition cycle has two layers of loops. The outer loop consists of information recognition and behavior adaptation, which are directly coupled with radio domain knowledge. The inner loop is a machine-learning loop where artificial intelligence methods are tailored and combined for a general solution making and self-learning. This two-loop cognition cycle matches the two-layer hierarchical machine learning structure. Recalling the Egg Model in Figure 2.2, the outer layer consists of radio domain-specific operations; while the inner layer is a general machine-learning core.

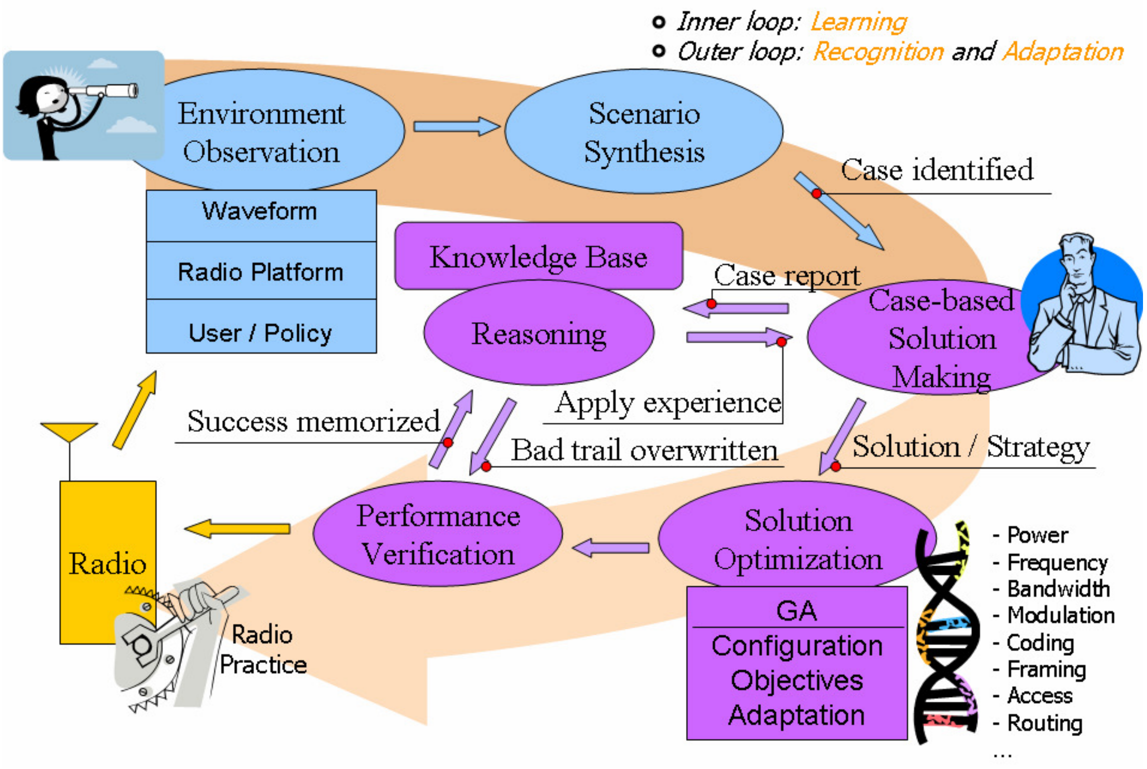


Figure 2.6: Cognition cycle block diagram

The full cognition cycle (1) collects environment information from the recognition modules, (2) synthesizes this information into parametric models for scenario representation, (3) compares this scenario with remember knowledge and decides whether to directly apply a previous solution, adapt a previous solution, or develop a new solution, (4) estimates the solutions' anticipated performance with formulated objectives, (5) applies the solution to the radio platform and monitors the actual performance, (6) compares the real performance with the anticipated performance, records the differences as lessons, and updates the knowledge base associated with this scenario-solution pair for future use.

Steps (1), (2) and (5) are carried by the outer loop that provides environment recognition and practice adaptation, together with the radio platform; and steps (3), (4) and (6) are carried by the inner loop that forms the machine learning cycle. Such a tiered structure has several advantages. First, it enables platform independence. General learning should be able to cooperate with different radio platforms and its intelligence should be applicable to various scenarios. A lower layer of radio domain-specific intelligence serves as middleware that interprets and abstracts the environment into standard representation and procedure for the learning core. This reduces the complexity of developing machine learning algorithms for the higher level intelligence by avoiding domain-specific considerations. Second, it simplifies system interface design. API efficiency and transparency are important to improve system efficiency, functional verification, operation management, and code reuse. Since the cognitive engine interacts with both software and hardware for a real-time performance response, the API between the cognitive algorithm and radio platform should be designed to be as straightforward as possible. A radio-domain-specific layer takes care of the physical meaning of all domain parameters, thus allowing the API to be simply script parsing. It also supports parallel software development and maintenance. System upgrade and porting become flexible by allowing both layers of cognition to be under parallel development. This feature is especially important for applying software defined approach for radio platform design.

In the cognition cycle, the outer loop serves the inner loop. It observes the environment and reports derived information like waveform features, interference and propagation channel characteristics, user service preferences, service policy and spectrum regulations, to the machine reasoning core. It also refines the solution from the reasoning core to improve performance, and formulates adaptation instruction, according to the solution, to feed the radio platform for action. The outer loop can be viewed as a lower-layer of cognition directly on top of radio application domain. Its intelligence level only guarantees (1) that the environment observation is sufficient and accurate for the machine reasoning and (2) that the adaptation exactly follows the instructions from the decision maker. The design of the radio environment recognition portion is detailed in Chapter 3.

Generally speaking, learning is a cyclic process which consists of making decisions based upon the awareness of the environment and the available knowledge, applying them as instructions for current (and future) behaviors, then observing the results of practice and educating itself by updating the knowledge base for future practice. This machine learning approach is commonly referred to as *knowledge based learning,* and the knowledge is evolved through the iterative reinforcement learning processing [39]. The inner loop, the learning loop, follows this learning approach. It synthesizes the problem scenario interpreted from the lower layer recognition and the performance objectives interpreted from the user service demands, and then search its database to associate relevant knowledge for solutions suitable for this interpreted situation. The reasoning process may use pre-loaded decision principles or accumulated experience from past practice, or both. When facing novel or unknown situations where a suitable knowledge instance is difficult to associate, a generally viable (but not optimal) solution can be further adapted with an evolutionary algorithm to fit such novelty; or a completely creative solution can be constructed through evolutionary search for a totally unknown situation. With rationality and creativity combined in solution making, the solution made and then the action taken have a higher probability of success facing an unknown situation, and meanwhile the action to take can be guaranteed correct facing a known situation. The performance of practice is evaluated and then reported to the learning core as the lesson associated to the deployed solution. In such a way, the knowledge is updated through experience.

Specifically, AI techniques are applied in both the outer and inner loops. Statistical pattern recognition and feature clustering methods are designed for radio environment recognition; the multi-objective genetic search method is designed for novel solution search to guide the radio's adaptation. In the inner loop, case based reasoning (CBR) is the primary solution making method, and is aided by genetic search to enhance solution adaptability. AI for outer loop is explained in Chapter 3, and AI for inner loop is detailed in Chapter 4.

Knowledge is the key in machine learning. It can be divided into two categories based on different sources. One type of knowledge is experience that includes situations, actions taken, and their consequences; the other consists of pre-set principles and non-adaptive examples. For machine learning system, the first type knowledge is achieved from self-practice, just like in humans, while the second type can be simply pre-loaded rather than through the long-term education that humans require.

For the CWT^2 cognitive radio system, the knowledge is implemented as a relational database, called the Cognitive Radio Knowledge Base (CRKB) [26]. CRKB is a meta-database that consists of several sub-databases, such as the radio environment map (REM) [40] for environment awareness, user service knowledge for performance objectives, case base knowledge for scenario-solution association, radio resource knowledge for solution boundary, and regulation knowledge for legality and security verification. As stated in Section 2.1.3, the CRKB concept matches the functional view of cognition. It can be flexibly implemented in a distributed way, unifying the cognitive behavior of both CR nodes and cognitive networks.

2.5 Summary

Chapter 2 presents the overall CR system design. It starts with the introduction of CWT^2 general CR solution with a set of definitions related to radio cognition. Then an innovative CR functional architecture is defined. The CR solution contains a software package of intelligent algorithms to support cognitive wireless link behavior and services. A platform independent radio interface is defined in the CR architecture to allow these intelligent algorithms to interface with radios having different hardware implementations. Besides, this CR architecture also features open-structure, modular design, so that it can be implemented in a distributed manner, which makes it suitable for network oriented cognitive wireless operations.

CR research is interdisciplinary, i.e., the CR system design depends on a wide range of background knowledge, which is partitioned to three major domains, radio, user and policy , facilitate the open-structure modular design for the cognition system. With a focus on the radio domain, the cognitive algorithm software system, the cognitive engine (CE) is defined. It includes key cognitive functionalities including environment recognition, solution making, and practice adaptation. Taken together, these form a complete machine learning cycle. A cognition cycle is defined as the soul that controls the CE operation. The cognition cycle is the core of the CWT^2 CR system. It has a two-loop hierarchical cognitive processing flow, which matches the CR hierarchical intelligence architecture. Machine learning is based on the application and accumulation of the knowledge. The knowledge for CR system is defined and formulated in an object oriented meta-data structure, and carried by XML scripts as the universal standard interface for information transfer.

Chapter 3: Radio Environment Recognition

3.1 Radio Environment Recognition Definition and Scope

3.1.1 Radio Environment Recognition Definition

Awareness is the first step toward cognition. There are two key questions related to awareness: the first is what environment information the CR needs to be aware of, and the second is how to enable the radio to sense the environment and achieve the required awareness.

Although there are three major domains that form the complete environment of CR technology, we focus our interest on the radio domain since it contains the primary technical challenges of wireless communications and radio engineering; while the policy and user domains are more concerned with policy and market strategies. Following the set of definitions given in Section 2.4.1, We provide additional definitions specifically about the CR's recognition capability in the radio domain here.

(1) Radio domain

The radio domain consists of the radio environment and radio platform. Although people would argue that the meaning of "radio" domain is vague, there are advantages to segmenting the entire CR technology background into three domains, radio, policy, and user domain for the convenience of domain analysis. Therefore, the radio domain in our definition can be simplified to contain two major groups of technology-based domain knowledge, the radio environment and the radio platform. The first group includes wireless air-interface related information, like communication waveform features and propagation channel characteristics. The second group is the information about the radio platform which provides the required processing resources. Such information includes power consumption, processor clock rate, filter settings, codec configuration, etc.

(2) Radio Environment

The radio environment consists of the wireless air interface related knowledge, such as radio waveform format and channel propagation characteristics. Such radio environment definition is necessary to describe what waveforms are in the air and how they behave at their particular radio frequency.

(3) Waveform

Waveform is defined as a super set of PHY parameters describing the format of a communication signal (PHY) and its related processing protocols (MAC, LLC, Net, etc.). This parameter set completely defines the wireless method of transceiving information between two communicating nodes. Such definition conforms to the waveform definition of the Software Communication Architecture (SCA) [41]. Such a waveform definition supports the standardization and portability of software defined communication applications.

(4) Sensing

Sensing functionality consists of Energy Detection and Channelization. As the first step in the recognition process (See Section 2.4.1.), the sensing of the radio environment consists of two stages of operation according to different levels of awareness requirements. Energy detection provides an overview of the spectrum power across a certain Radio Frequency (RF) range of interest; channelization is performed to extract a certain relatively narrowband portion of the spectrum occupied by a waveform, to prepare for the further investigation of this waveform's detailed format, which is the waveform classification.

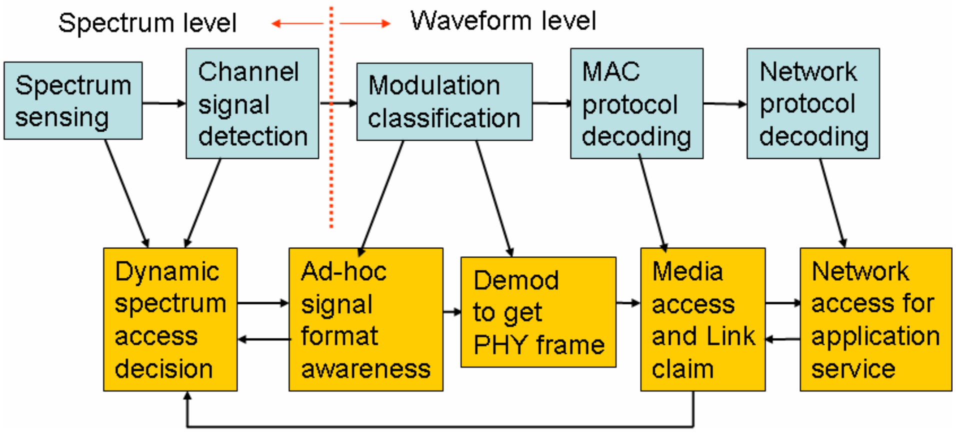
(5) Classification

The classification defined for radio environment recognition here is a statistical separation procedure in which the radio environment facts are modeled into quantitative characteristics, called features, and then grouped by such features. Statistical pattern recognition and feature clustering methods are required by the noisy nature of the radio environment.

In radio domain awareness, this dissertation focuses on the design and implementation of the recognition for radio environment. In a practical CR system, radio platform knowledge is preferred to be loaded into CR's knowledge base rather than accumulated through an ad-hoc recognizing process. Radio platform knowledge design is described in Chapter 4.

3.1.2 The Two Levels of Radio Environment Knowledge

The radio environment knowledge can be separated into two groups that lead to two awareness levels in supporting the radio domain cognition, shown in Figure 3.1.



Spectrum level
Waveform level
Spectrum sensing
Channel signal detection
Modulation classification
MAC protocol decoding
Network protocol decoding
Dynamic spectrum access decision
Ad-hoc signal format awareness
Demod to get PHY frame
Media access and Link claim
Network access for application service

Figure 3.1: Radio environment knowledge for cognitive behavior

The first level is the spectrum energy profile including the location and power of existing signals across a certain range of frequency. Such knowledge of the spectrum overview makes the CR aware of the energy presence and occupation pattern in time at the frequency of interest, thus enabling itself to pick the right channel, the right power, and the right time of using the spectrum to avoid interference. In other words, general spectrum management algorithms, like Dynamic Channel/Spectrum Allocation (DCA or DSA), can be applied with this level of awareness. Spectrum overview is obtained through frequency sweep and energy detection at across the frequency range of interest, detailed in Section 3.2.

The second knowledge level includes knowing the format of the waveform at the channel of interest, and modeling the propagation channel in that spectral region. It comprises a step-by-step sensing and classification procedure, in which different levels of information are extracted at different stages of the receiving process. Such knowledge of the observed channel waveform makes the CR understand who else is out there and how to communicate with them. Such knowledge is obtained along with the signal receiving process. It is system-level design challenge that requires a hierarchical signal processing system from RF tuning to baseband decoding so that different levels of waveform parameters are obtained gradually.

Knowledge of the propagation channel guides the receiver to minimize the waveform distortion for more accurate signal observation; meanwhile it also helps to adapt the signal transmission through the channel. Propagation channel recognition and equalization design become more complicated in CR because there may be little prior channel knowledge available, especially when the target waveform is unknown. Thus highly-adaptive or even “blind” channel modeling techniques are needed to clean the waveform in an unfamiliar or unknown channel.

The hidden node problem is difficult to solve with local self-sensing, but collaborative sensing using multiple network nodes can greatly alleviate this problem. There are two ways to deploy a collaborative sensing system: either a specific sensing network can be set up in addition to the service network, or else two networks may be mergted by integrating sensing capability into service nodes. The first choice has interoperability issues between networks, while the second choice is directly enabled using cognitive radios as network nodes. Collaborative sensing coordinating algorithms are outside the scope of this dissertation.

3.2 Spectrum Sweep and Energy Detection

Conventional signal detection techniques include matched filtering and energy detection in either the time or the frequency domain [42]. However, matched filtering assumes prior signal knowledge so that decision theoretic detection can approach optimal performance against an Additive Gaussian White Noise (AGWN) channel. Energy detection in the time domain squares the incoming signal samples and determines the energy presence according to a certain amplitude threshold. Although computationally efficient, this time domain method lacks flexibility in spectral differentiation and is susceptible to channel noise and distortion. One example is the Received Signal Strength Indication (RSSI) used in conventional mobile radio receivers. In the frequency domain, first-order Power Spectral Density (PSD) has been widely used for various stochastic signals and systems [43], although it is also generally noise sensitive due to its first-order mathematical nature. Second-order Spectral Correlation Density (SCD) can be calculated to track a stochastic signal's cyclostationarity properties as the detection criterion [44]. This method has been getting more attention due to its two distinctive capabilities for suppressing white noise and separating different cyclic components overlapped in spectrum [45-47] . However, the computational cost of bi-frequency domain correlation could be prohibitive for real time processing [48]. It is unnecessary to replicate a literature review for classical spectrum analysis techniques. S. M. Kay's tutorial paper [43] serves as the best survey on this topic, and an update of Kay's survey and more detailed introduction of both first-order and second-order spectral analysis comes from S. L. Marple [49].

The line between energy detection and channel waveform classification varies with target application scope and system design requirements. When the target spectrum is filled by signals with known formats that can be differentiated by their spectrum properties, energy detection and waveform identification can be accomplished at the same time. Another special case is that when the waveform is known to have a pilot component, which typically has a spike with sufficient energy to be easily detected, then the waveform is also identified by recognizing this pilot "signature". Under normal conditions without auxiliary waveform knowledge, spectrum and waveform recognition are two separate steps, where spectrum sweep only gives a frequency-power profile and waveform recognition takes over a channelized signal of interest (See Figure 3.1 above.).

The spectrum sensor system and applied detection methods should be designed to match the target application. In the CWT^2 CR system's spectrum sensor module, spectrum is separated by multiple blocks according to the radio front-ends' linear tuning range, and PSD based energy detection is applied with noise adaptive threshold control. Time-domain averaging is applied to the resulting PSD, which is based on a modified Welch periodogram adaptive to the incoming signal SNR. With no prior waveform knowledge and general SNR conditions, the Welch periodogram provides the best performance [50].

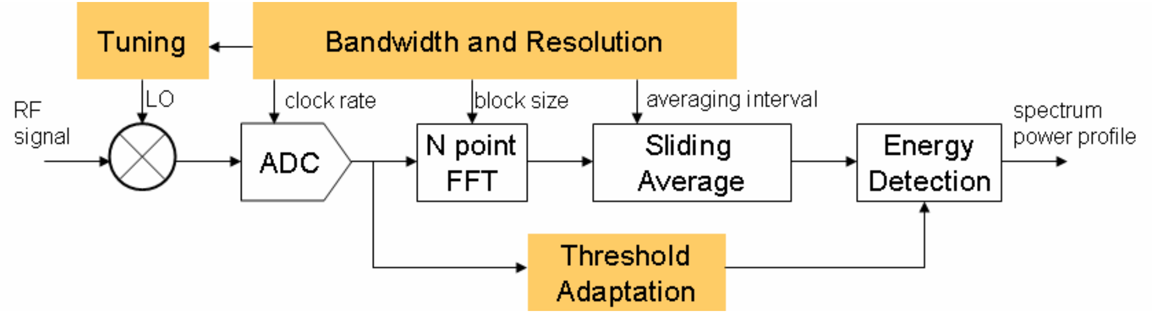


Figure 3.2: PSD based energy detector

Figure 3.2 shows the block diagram of the spectrum energy detection system. Taking the digital signal samples from the Analog-to-Digital Converter (ADC), the PSD is calculated by a Fast Fourier Transform (FFT) [51]. To achieve flexibility in balancing performance and processing time, the spectrum sweep is separated into two steps, a whole band sweep with one or more coarse FFTs and then FFTs with finer resolution in the band of interest, if needed. The target frequency range is divided into multiple blocks according to both the front-end linear bandwidth and frequency resolution required by FFT. In both steps, FFT block size is adjustable for resolution and processing time requirements. Time-domain windowing is applied to reduce spectrum leakage before the FFT. Normally multiple FFTs with moderate block size are preferred. This is not only because processing time is saved, but also because block based temporal averaging can be applied to smooth the noisy result. The reference noise level is constantly measured and updated via a sliding average. The system operation flow chart is shown in Figure 3.3. As shown in the figure, there are two loops; the outer loop is for multiple RF tunings and the inner loop is for multiple block FFTs. The final output is a spectrum power profile described using standard Extensible Mark-up Language (XML) (see Chapter 4) that can be distributed and interpreted by other processing units in the CR system.

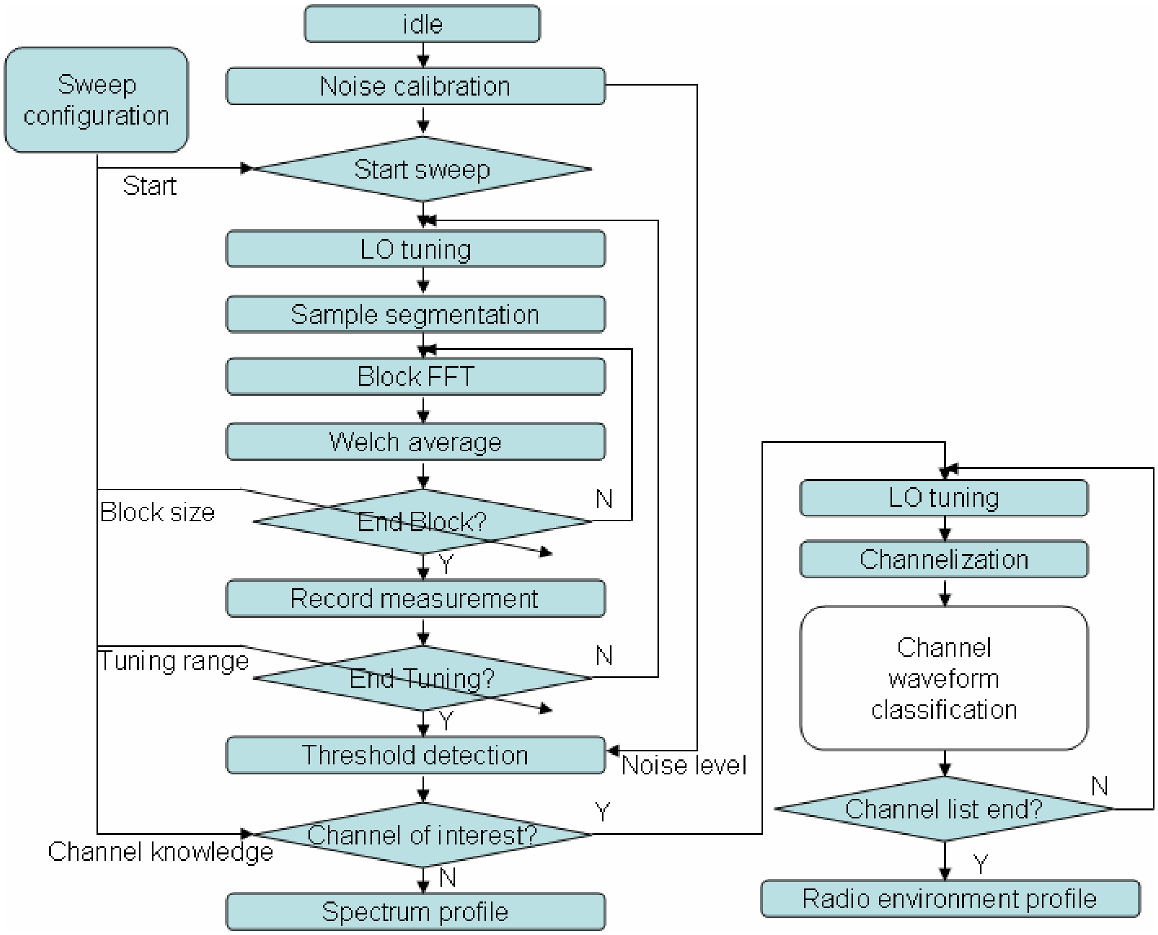


Figure 3.3: Spectrum sweep and signal classification system flow chart

The energy detect system can further perform channelization to extract samples from certain channels of interest. There are two ways of configuring the channelization. One way is that the signal is channelized based on reference information like standard channel bandwidth and carrier frequency from an existing radio environment knowledge base; the other is that the channel location and bandwidth are purely determined by measured power and threshold setting. Once the signal is channelized, it is handed to the waveform recognition module for identification. Then the final result contains both the spectrum power information and also waveform information for the channels of interest. Taken together, these compose a complete radio environment profile. This profile shares the same XML format with spectrum profile but with extended information in the channel data fields. Therefore there is only one data interface format between the sensing modules and the learning core in the CR system.

Because FFT calculation is based on complex digital samples, the resulting PSD does not have asymmetric layout, therefore, single RF tuning covers both the lower and upper side of the spectrum around the center frequency. An example PSD of a complex Quadrature Phase Shift Keying (QPSK) modulation signal is shown in Figure 3.4; this QPSK signal has a 20k baud rate and a 0.35 pulse shaping roll-off factor. It is generated by the CWT^2 waveform framework with Universal Software Radio Peripheral (USRP) (See Chapter 6.), and collected by an Anritsu™ spectrum analyzer. Note that there is a several-kHz carrier frequency offset between transmitter and receiver.

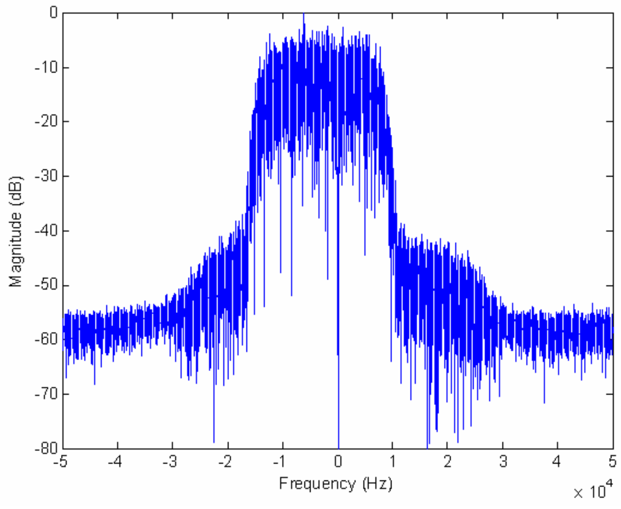


Figure 3.4: PSD of a complex QPSK signal, 20k baud rate, centered close to DC

Most modern ADCs provide complex digital output; if not, a frequency domain Hilbert transform can be implemented before complex PSD calculation. The block diagram is shown in Figure 3.5. A real signal, *x(t)* is first transformed to frequency domain, *X(f)*, and then passed through the Hilbert transform, *H(f)*:

$$H(f) = -j \cdot sign(f)\,. \qquad (1)$$

The Hilbert transform shifts the negative frequency components by 90^{o} and positive frequency by -90^{o}. Then the resulting signal $X^{T}(f)$ is inverse transformed to the time domain as $x^{T}(t)$. Then the complex analytical signal can be obtained by

$$x_a(t) = x(t) + j \cdot x^{T}(t)\,. \qquad (2)$$

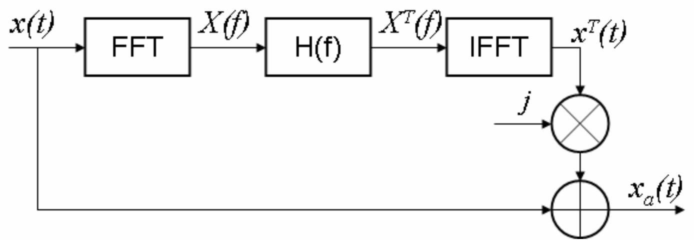


Figure 3.5: Digital frequency domain Hilbert transform block diagram

The analytical signal model is very important to the digital signal processing, because the complex signal spectrum does not contain the image spectrum. This provides a great convenience in the digital implementation of frequency conversion and filtering, especially for the synchronization loop design. The detail of theoretical framework of the Hilbert transform is available from [52].

3.3 Signal Classification Overview

3.3.1 Signal Classification in Waveform Recognition

As defined in Section 3.1.1, the waveform includes both signal parameters and protocol settings; thus the corresponding recognition procedure contains the classification of both parts. As shown in Figure 3.6 below, waveform recognition requires a processing chain from RF to baseband so that different levels of information are obtained gradually, often the recognition at the next stage relying on the result from the previous one. Within this chain the signal classification is the starting point as parameter recognition starts from PHY layer of the incoming signal. Within the signal classification process, key parameters from the radio frequency to the modulation scheme need to be identified step by step. Among these the modulation classification is the most challenging part. Modulation is the key information that a receiver needs to know before it can successfully extract correct information contained (or modulated) in the incoming signal.

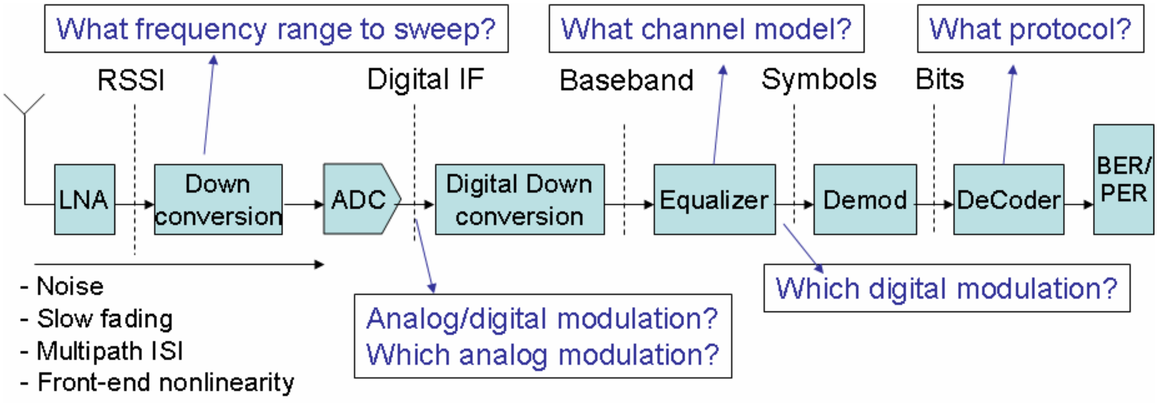


Figure 3.6: Waveform recognition processing chain block diagram

In order to understand a received signal, the modulation related PHY parameters need to be recognized first to enable further signal processing and recognition. Once modulation is classified and the information bits are demodulated, the subsequent recognition tasks are taking care of frame and packet format check and decoding largely based on straightforward table lookup. The modulation classification is the key of waveform recognition; it is also the focus of this chapter.

3.3.2 Brief Literature Review of Signal Classification

Signal analysis and classification has been a long-term research topic in communications. A lot of analyses and systems are proposed on this topic. In terms of mathematical model and computation technique, they can be roughly separated into three major approaches. The first is based on the Maximum Likelihood (ML) principle, where matched filtering detection is applied on certain features of the input signal. Higher order correlation features that are used for asynchronous ML classifiers typically require certain prior knowledge [53]. And its optimal performance is based on signal coherence [54]; to make the system more realistic, asynchronous ML classification is also proposed [55] at the computational cost of 2nd and higher order correlation for feature extraction. Optimal classification rules with ML detection are investigated in [56], while the classification result is limited to BPSK and QPSK. The log likelihood criterion is mostly applied for detection. And most of the ML based classification systems available in literature limit their results to within a small group of modulations.

The second group is based on extracting features from a target signal and using feature classification methods to differentiate modulations. Zero-crossing was found effective for non-coherent classification, but is sensitive to Signal to Noise Ratio (SNR) [57]. For envelope based classification, a novel envelope extraction method is proposed by [58] which is faster than the traditional Hilbert transform method. Many proposed systems use histograms. Other approaches include using histograms of the phase, envelope, and instantaneous frequency of the target signal [59-63], which are all first-order features in the time or frequency domain. For the classifier design, using an Artificial Neural Network (ANN) is the most popular choice for a pattern recognition approach; A hidden Markov Model (HMM) with M-th order FFT is used for classifying M-ary Frequency Shift Keying (MFSK) modulations in [64]. The theoretical basis is primarily built by [65] and [66]. Many feature extraction designs are proposed in the past a few years [67-69]. More complicated ANN and feature set designs are also proposed. A hierarchical ANN is proposed by [70] to replace a bulky full-connection network. With such hierarchical network structure, a further comprehensive feature set (with 31 features) is defined in [71] to achieve good performance. A good survey of signal modulation classification using pattern recognition approach is provided by Nagy [63]. However, the related training time could be prohibitive for on-line reconfiguration. .

The third group is using the second or higher order spectrum correlation properties as the features to classify the modulation schemes. The theoretical framework is largely formed by Gardner's book [72] and several classic papers [44, 73, 74]. In [47] spectrum correlation properties are used to separate overlapping communication signals which have different cyclic features. Recent works are using spectrum correlation as the modulation feature, and the performance is robust against very low SNR [45, 46, 75]. ANN [46] and HMM [75] are also used for the feature classification.

3.3.3 Signal Classification Design Challenges

Today's wireless communications is based on fixed communication standards. By standard, waveform implementation is a set of pre-defined working modes. Thus the key design challenges at the receiver side, such as carrier synchronization, channel equalization, demodulation and channel decoding, are reduced to optimization problems for fixed waveform formats. In other words the key parameters of the input waveform can be anticipated, and thus a decision-theoretic approach is implemented along the receiving procedure. The performance can be optimized through maximal-likelihood detection against AWGN [51]. For example, one major issue in conventional receiver design is synchronization [76]. Specifically, most carrier phase lock and demodulation designs assume and use the baseband signal feature which is defined by modulation; and for digital modulation most symbol timing algorithms know the symbol rate (baud rate) and symbol pulse shape in advance [42].

Obviously when the CR needs to recognize an unknown waveform, many key signal properties, like carrier frequency, signal bandwidth, symbol rate and modulation scheme, need to be identified by the radio itself. This implies two design challenges. (1) The signal classification needs to identify the key parameters to enable successful signal reception, and carrier synchronization is the key. The signal classification's major task is to identify the incoming signal's modulation scheme to guide the synchronization module. The difficulty is that most modulation-sensitive information like complex phase change and constellation is at baseband and therefore only available after achieving carrier phase lock. Therefore, the modulation classification task needs to solve a "chicken or egg" dilemma. (2) The absence of prior signal knowledge makes the conventional standard-specific design not suitable for the cognitive radio receiver. In CR, the transceiver should be able to handle various types of waveforms. Specifically, a new receiver structure is needed that can synchronize and decode different signal formats. The challenge is mostly in the synchronization design. The synchronizer should have a general looping structure to track incoming signals with different modulation schemes, and more importantly, be real-time reconfigurable to adapt its nonlinear operation and looping bandwidth according to the modulation related information from the signal classifier module. Once the signal carrier is synchronized, the signal classifier becomes able to extract more signal information at the baseband. The relation between synchronization and modulation classification is illustrated in Figure 3.7.

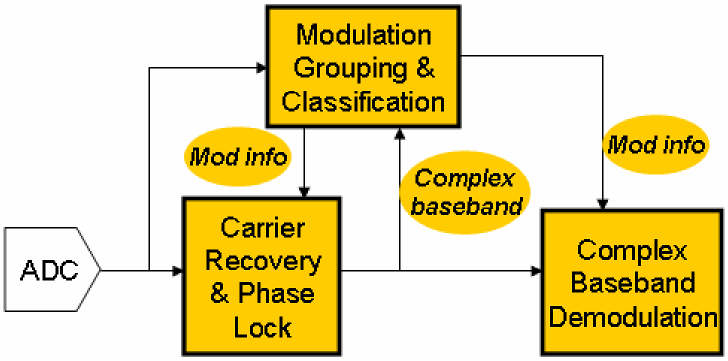

Modulation Grouping & Classification
Mod info
Complex baseband
Mod info
ADC
Carrier Recovery & Phase Lock
Complex Baseband Demodulation

Figure 3.7: Carrier synchronization and modulation classification in CR

The input signal at the receiver has experienced attenuation and possible nonlinear effects in the propagation channel. Conventional receiver design typically assumes a certain channel model and implements a corresponding equalization algorithm [77, 78]. In CR applications, the prior knowledge of channel model may not be definite or even may not be available. Under such condition, the channel also needs to be recognized and compensated. This similarly leads to two design challenges, channel recognition and equalization adaptation design.

According to the public safety communication application scope, a flat AWGN channel is assumed in the current research stage. For complicated fading and time-varying channels, general channel model is needed to track different propagation characteristics, and highly-adaptive or even blind convergence algorithms are needed for this case. This is out of the scope of this dissertation. For interested readers, there are a couple of good tutorial papers [79, 80], and some other papers using the most popular modulus-based algorithm [81-88] or higher order statistics [89].

3.3.4 Feature Based Classification Approach Introduction

There are two major design approaches for modulation classification, maximal likelihood and feature classification. As stated in Section 3.2, maximal likelihood leads to a matched filter classification structure. Theoretically such a structure is able to approach optimal detection performance against white noise, but under the assumption of ideal signal coherence. And the detection performance reduces significantly when coherence is slightly disrupted [90], while in CR the signal coherence may not be available before the modulation scheme is classified. A matched filter typically requires certain prior knowledge of the signal parameters. Such knowledge is necessary to evaluate the "matching" on these properties of the incoming signal. Unfortunately these parameters are at baseband after synchronization, where the task for modulation classification becomes trivial. Therefore a matched filter approach is better for a standard based receiving structure rather than for the CR.

However, for some specific waveforms with distinctive features, a matched filter will be a better choice because maximal likelihood can be applied at time or frequency domain before carrier sync. For example, waveform with special pilot can be easily recognized with a spectrum match. And under a normal receiving mode where signal format is known, the matched filter is always preferred due to its best performance and efficiency.

In feature based classification approach, feature parameters are extracted from the incoming signal and used as the criteria for the decision. The definition of the signal features is flexible as far as it helps in differentiating modulation schemes. Required features can be extracted at any stage of the signal processing chain. A general feature based classification system block diagram is shown in Figure 3.8.

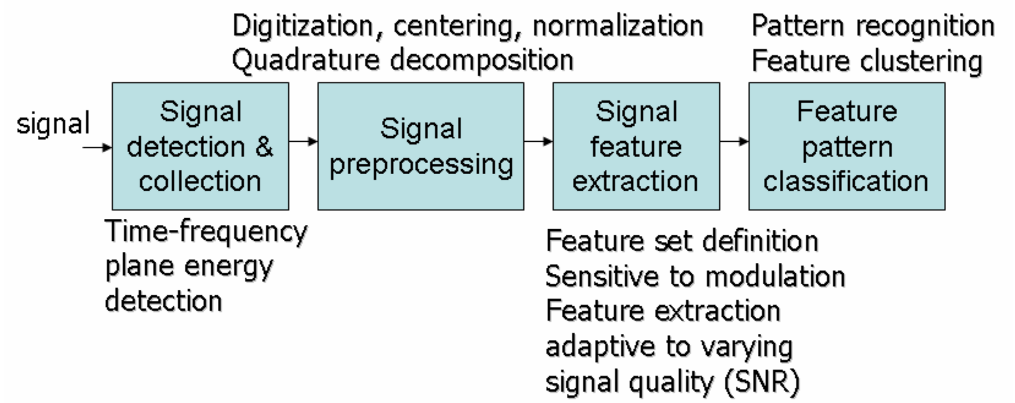

Digitization, centering, normalization
Quadrature decomposition
Pattern recognition
Feature clustering
signal
Signal detection & collection
Signal preprocessing
Signal feature extraction
Feature pattern classification
Time-frequency plane energy detection
Feature set definition
Sensitive to modulation
Feature extraction adaptive to varying signal quality (SNR)

Figure 3.8: Feature based signal classification system block diagram

In Figure 3.8, the input signal is first detected and collected. Then it follows a series of steps of signal pre-processing like digitization, amplitude centering and normalization. Quadrature decomposition may be needed to convert the real signal to the complex domain. The preprocessing is to clean the signal sample in order to improve the signal feature quality. Then a certain feature set is defined and extracted from the collected signal. The modulation scheme is determined by the classification of these features.

3.3.5 Signal Modulation Feature Dimensions Introduction

Feature based modulation classification is a two-step task including feature extraction and classification. A good feature set is the key to successfully classifying different modulation schemes. A communication signal typically can be described in three mathematical dimensions, time, frequency and complex vector space. Accordingly, the features related to the signal's modulation scheme can be defined in these three dimensions.

Temporal features

A modulated signal can be represented in complex format as:

$$s(t) = Re\{A(t)g(t)exp(j2\pi f(t) + \varphi(t))\} \tag{3}$$

where $A(t)$ is the amplitude, $g(t)$ is the symbol pulse (only existing in digital modulations), $f(t)$ is the frequency and $\varphi(t)$ is the phase. All temporal signal features are derived from these parameters in the temporal equation, and most of such features are the statistics of such time domain parameters. For example, complex envelope related statistics can be used to classify frequency modulation signals [91]; zero-crossing statistics of the amplitude swing can be used to estimate the center frequency of the real signal [57]; higher order moments of temporal parameters are also used for a larger classification set [92]. Some example time domain modulation waveform representations are shown in Figure 3.9.

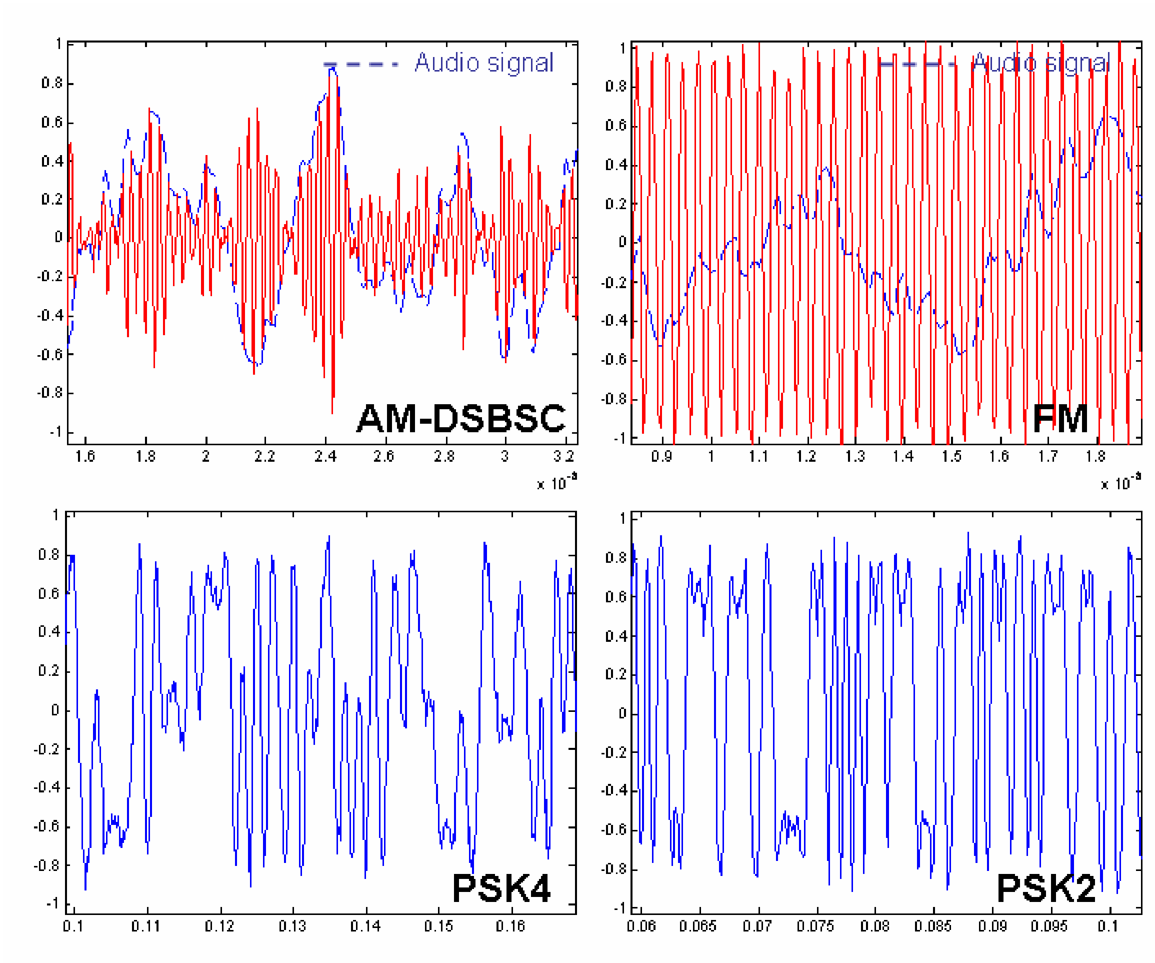


Figure 3.9: Example modulation signals – time domain

Temporal feature-based modulation classification directly uses the collected signal sample stream for feature calculation. Most of extracted features are first or second-order statistics, straightforward and easy to implement in real-time block processing. However, these features are typically sensitive to noise and distortion. The trade-off between feature robustness and computational complexity is the key in defining temporal features.

Spectral features

Compared to temporal features, spectral features are more stable against noise because they are extracted by correlation and statistical averaging. The classic spectral analysis is based on a signal's first-order spectrum statistics, i.e., the power spectral density (PSD).

For the example analog and digital modulation signals shown in Figure 3.9, their PSD plots are shown in Figure 3.10. Both PSK and PSK2 modulation signals use a 0.35 roll-off value for the pulse shaping filter (psf).

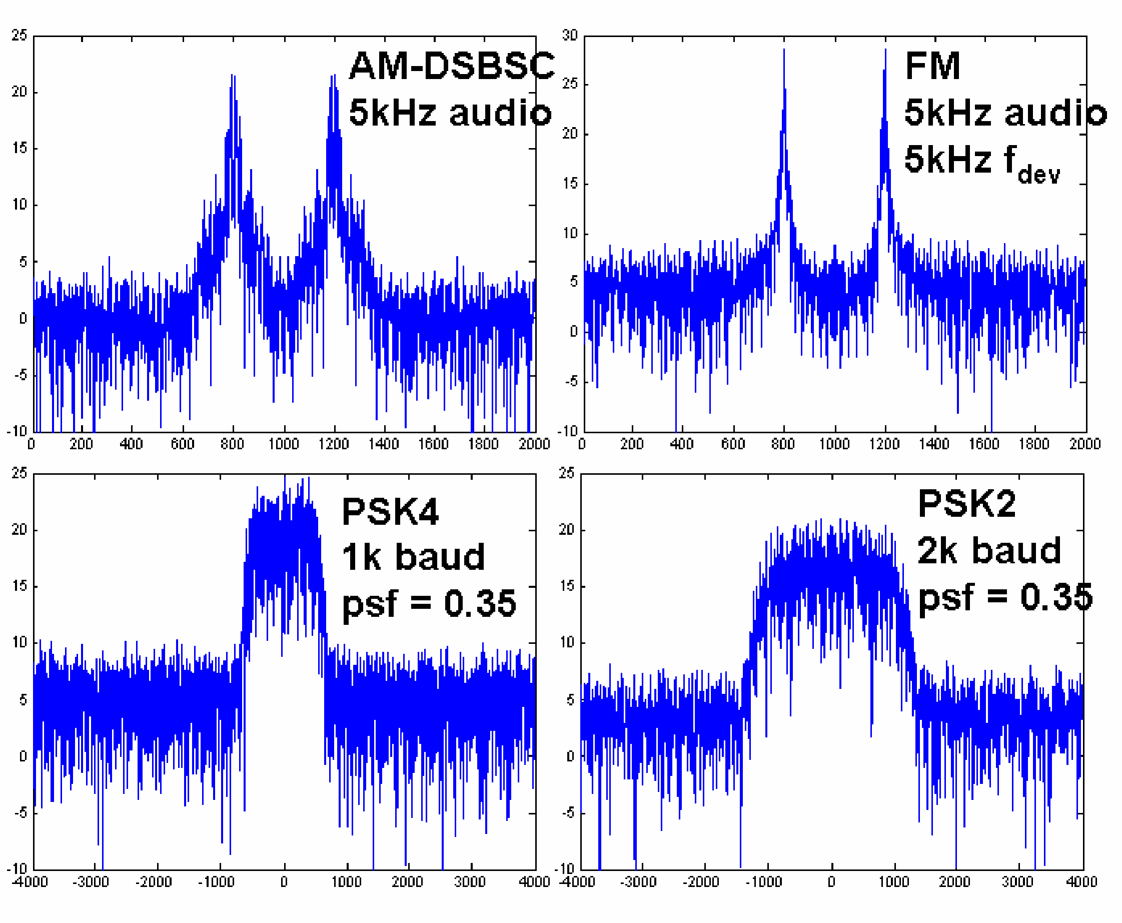


Figure 3.10: Example modulation signals – spectral dimension

PSDs are commonly calculated in the digital domain using the FFT. The block of samples can be windowed to reduce spectrum leakage. Since PSD is first-order spectrum correlation, it is usually noisy; therefore various averaging methods have been applied to achieve a balance between time domain averaging for statistical stability and the resulting frequency domain resolution penalty with fixed number of samples [43]. The most widely used time domain averaging method is the Welch periodogram to estimate the signal PSD [93]. The Welch periodogram method is used in the CWT^2 CR spectrum energy detector with the enhancement of overlapped block averaging on the periodograms to improve PSD detection stability. (See Section 3.2.).

The calculation of PSD and its related first-order spectrum statistics is relatively efficient due to the FFT-based implementation. However, it does not provide too much insight into the signal's modulation scheme. The modulation operation has a multiplication relationship between carrier and modulating data signal. It implies a second-order periodicity, which is called cyclostationarity [72]. All communication signals have a cyclostationary nature, i.e., their statistical properties vary periodically with time. Such second-order (cyclostationarity) spectrum statistics reflects the hidden modulating mechanism. It is extracted through the second-order frequency correlation and the process is robust against severe noise and moderate distortion [44]. Higher-order spectral statistical calculations can also used to improve the separation between signals that have different cyclic properties. However, the huge computational cost makes higher-order approaches less attractive for on-line processing. For modulation classification, only the second-order cyclic feature is considered since most multiplication-based modulation mechanisms can be sufficiently identified by second-order cyclostationarity [73, 74]. To save computational cost, FFT-based spectrum correlation is commonly used for cyclostationarity calculation [46, 75]. In the CWT^2 CR system, second-order spectrum correlation is only used when the incoming signal is extremely weak or even below the noise floor. Most of the time first-order temporal and spectral features are enough for classifying a signal with at least moderate SNR, which are needed anyway to build a useful wireless links.

Complex vector space features

Vector analysis applies only for digital modulations whose complex symbol formats can be represented as constellations in the in-phase/quadrature (I/Q) plane. This provides a graphical insight into the underlying modulation schemes. Distribution statistics of the analytical phase of the complex symbol value can be easily calculated and matched to the modulation orders. Such a phasing feature can be very useful to differentiate between high-order quadrature, linear, digital modulations, which are typically very difficult to identify before carrier synchronization [29]. Although equivalent to time-domain representation, it is a symbol-level feature obtained after channel equalization and matched filtering. Therefore, it is more robust against noise and distortion than temporal waveform statistic calculations before the synchronization stage. Since every digital receiver has a quadrature I/Q differentiator to track the phase error [94], this is a by-product that we can use with no additional cost. Another advantage is that vector space is calculated at symbol rates with greatly reduced computational cost than at previous stages. Figure 3.11 shows the constellation of the example modulation signals.

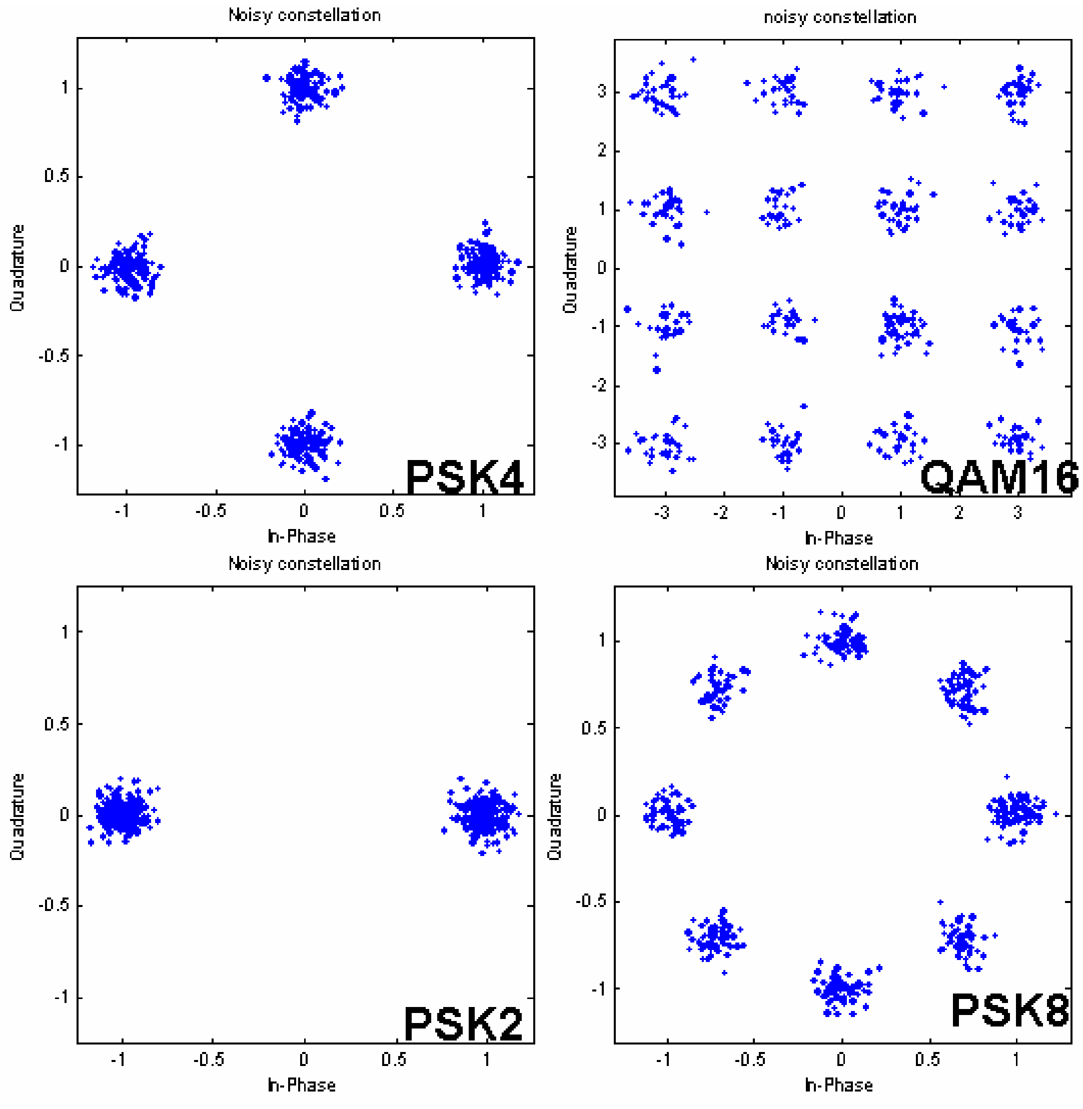


Figure 3.11: Example modulation signals – complex vector space, SNR=20dB

3.4 CWT^2 Modulation Classification System Overview

3.4.1 System Structure

It is difficult and not necessary to make a “universal” modulation classifier that can classify arbitrary signals. A practical modulation classifier design depends on the target signal modulations, target signal and channel quality, supporting radio platform structure, available processing resources, and application service requirements. In the CWT^2 CR node system, an adaptive signal classification system is designed, as shown in Figure 3.12.

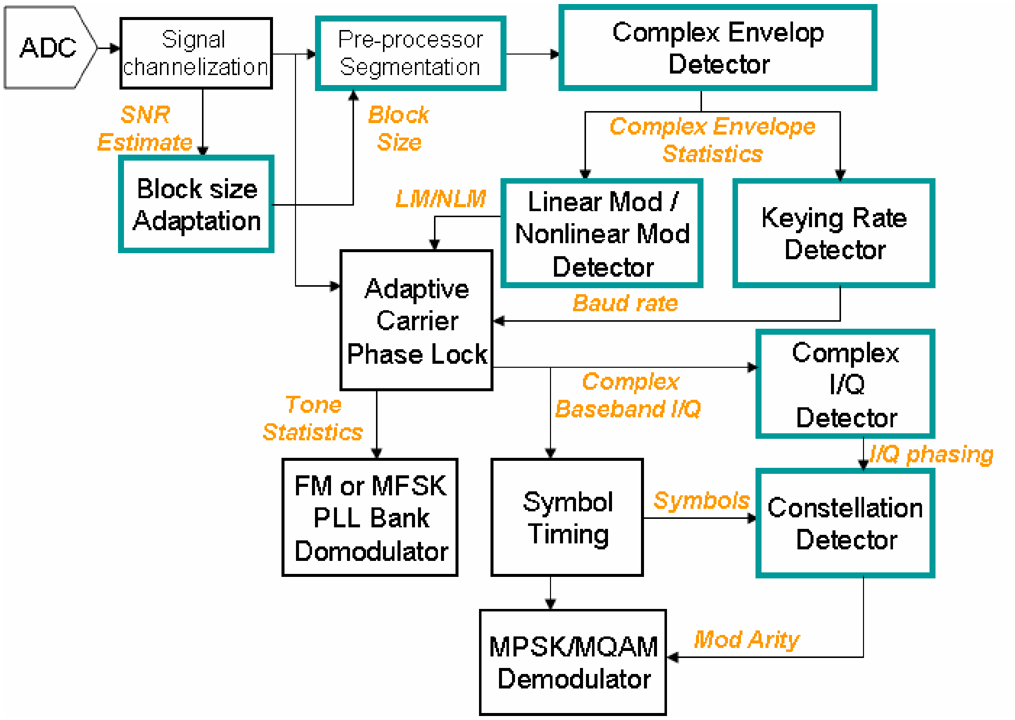


Figure 3.12: CWT^2 adaptive signal classification system block diagram

In Figure 3.12, the input signal is frequency down-converted to baseband. This signal is called complex quasi-baseband because it still has some small residual frequency offset for several reasons. First, in practical system, there is always some mismatch between transmitter and receiver carrier frequency; second, real oscillator's frequency will drift with time and temperature, especially with low cost, mobile handheld devices; third, the Doppler shift induced by terminal mobility could cause additional frequency offset. With the current radio front-end that the CWT^2 CR node is using, a typically frequency offset is less than 5k Hz, as shown in Figure 3.4. Such frequency offset is relatively small compared to the baseband modulation signal phase transition and thus can be synchronized by a phase locked loop.

The complex quasi-baseband signal is first segmented into signal blocks, centered and normalized. Then the following signal feature extraction and classification steps are all based on block processing. Aligned with genera receiver signal processing practice, the modulation classification system consists of two stages. The first stage is before carrier synchronization and the second stage is after.

At the first stage, complex envelope related features are extracted from the un-synchronized signal to make a coarse classification between different modulation groups, such as real or quadrature modulation, linear or nonlinear modulation, etc. The key is that such decisions should be adequate to guide a modulation-general carrier synchronization to achieve carrier phase lock for the classified modulation group. A quadrature multiplication structure [95] [96] is selected because the in-phase and quadrature branches can be flexibly reconfigured for both real and quadrature modulations. And with full digital implementation [97], the phase error detector and loop filter are reconfigurable for both linear and nonlinear modulations; the phase error detectors can also be configured as an FM demodulator; and the structure can further be replaced by a low-pass FIR filter for analog AM signals. Besides modulation grouping, for digital modulations, the keying rate should also be estimated to set the initial loop bandwidth of the carrier synchronizer. The keying rate can be directly estimated from time domain envelope based feature through FFT [98, 99], PSD shape characteristics, or through second-order spectrum correlation.

At the second stage after carrier synchronization, the task is to continue and refine the classification. Modulations with different orders within the same group need to be classified based on the features extracted from complex baseband signal samples. Before symbol timing, quadrature phase and amplitude transition statistics from complex samples can be exploited for differentiating different modulation orders, which generally accomplishes the modulation classification. However, the symbol level statistics at the output of the symbol timing block can further refine and verify the classification result.

3.4.2 Features of Two-stage System Structure

Such a two-stage signal classification system has three major advantages:
(1) Adaptive signal classification

Different modulation signals have their own modulating mechanisms that exhibit different characteristics along the receive signal processing chain, especially the synchronization that enables the demodulation. Some modulations have strong features before synchronization, while others are not easily separated until their complex baseband information is available. Therefore, it is more practical to treat different modulation signals at different stages according to their extractable features' sensitivity to the modulation scheme.

A two-stage classification system is structured surrounding the carrier synchronization. It is important to clarify that the first step of modulation classification is to extract enough information to guide the carrier recovery and carrier phase lock, so that the unsynchronized signal can be translated into baseband. After that more modulation information becomes available to continue with a second round of processing with newly available features, so that more modulation schemes can be identified, especially so that modulations with different orders within the same group can be differentiated. This is very difficult before complex baseband statistics are available.

The two stage system structure enables adaptive signal classification according to the target problem. The synchronization loop is configured according to the classification result from the first stage, and the feature set and their related extraction and classification methods are also adapted. In this way, a processing flow with a decision tree is dynamically constructed for the input signal sample space. This is especially useful for a practical application with a fixed set of waveforms and efficient in resource consumption for embedded implementation.

Instead of following a sequential order, two classification stages can work together to form a bootstrap process where the result from the second stage can feed back to the first stage and the synchronization block to change the carrier phase lock loop configuration for a better fit for the incoming signal. Therefore, they are helping each other to improve performance.

(2) Three-dimension hybrid feature extraction

Specifically for modulation classification, the choice of feature parameters should follow these basic guidelines:

(i) Feature should be sensitive to modulation scheme, but insensitive to the modulated data.

(ii) Feature should be insensitive to carrier frequency or time shift, i.e., to initial phase.

(iii) Feature should be insensitive to channel added noise.

(iv) Feature should be insensitive to phase jitter and fading.

Generally, modulation schemes have distinctive sensitivity to different feature parameters. Even for the same modulation, modulation-related characteristics exhibit different visibility at different feature dimensions. In the CWT^2 classification system design, signal features are defined across all three dimensions, temporal, spectral and complex vector space. The principle is to use the right feature to extract specific information that is needed at the specific processing stage.

Three types of signal features, temporal, spectral and vector space statistics are used to jointly determine the modulation of incoming signal. Generally first-order temporal features can convincingly identify analog and digital modulation groups, and simple digital modulation schemes. But this method may have problem in classifying higher-order modulations due to its first-order sensitivity to noise and distortion. However, with such knowledge the synchronization loop can achieve phase lock and the complex baseband signal is available. In second stage complex phase based temporal statistics are effective in identifying more digital modulation schemes; also the constellation pattern based on the symbol obtained after symbol timing is effective in identifying the modulation order.

A first-order FFT is always useful for estimating the IF carrier frequency, and it is also effective in estimating the keying rate to configure the loop bandwidth for carrier phase lock at the first stage. However, it is less informative in estimating baseband bandwidth with varying SNR. The analysis on first-order temporal-spectral statistics usually takes much less computation compared to the second-order, but it also bears the linear susceptibility to added noise. This, however, doesn’t detriment the applicability of first-order methods for modulation classification because when SNR is low, modulation signals typically can not be successfully demodulated even their modulation schemes can be identified with higher-order cyclic features detection via tedious correlations. On the other hand, when signal classification is not directly used for normal link demodulation, but for energy surveillance or interception, higher-order statistics are perfectly suitable for off-line analysis on recorded waveform samples. The calculated FFT vector at the first stage can be directly slide-correlated for cyclostationarity calculation.

The feature extraction and classification design at each stage are detailed in the following sections; see also [21, 29].

(3) Different classifier for different feature sets at different stages

As stated above, different features from three dimensions are jointly used at each stage to identify the modulation. At different processing stages, the different features are combined and exhibit different feature space characteristics such as complexity, noisiness, etc. Different identification methods are applied to suit different feature space, and they are also designed to facilitate on-line processing. Figure 3.13 shows the identification algorithms that are designed at the different stage of the modulation classification system.

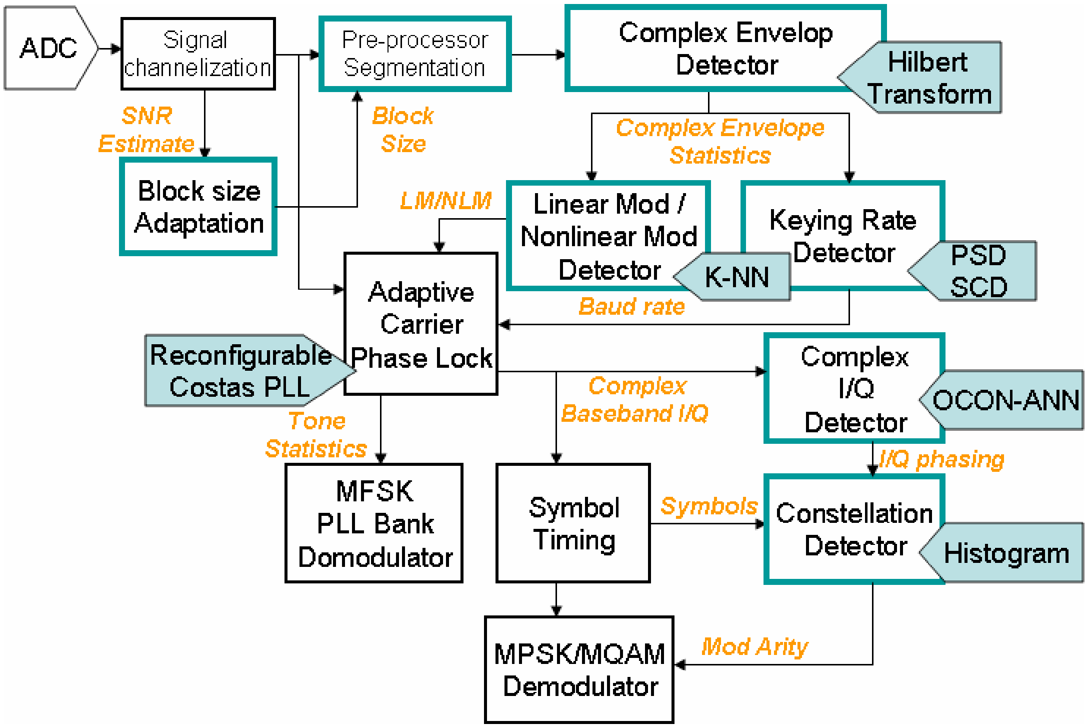


Figure 3.13: Classifier algorithms in classification system

The feature space before carrier synchronization is relatively simple so the K-Nearest Neighbor (K-NN) algorithm is designed for the feature clustering. It is also used in keying rate detection for a standard modulation lookup if some standard modulation knowledge is available. At baseband, more informative features are available and the feature space is relatively complicated and multimodal with different input signal scenarios. Therefore, a statistical pattern recognition approach is applied and a modular One-Class-One-Network Artificial Neural Network (OCON-ANN) is designed for feature classification. For constellation match at symbol level, a complex histogram can be used to identify the modulation orders according to the number of spikes. Both of them are detailed in the description of each classification stage.

3.5 Stage 0: Adaptive Segmentation and Preprocessing

The complex quasi-baseband signal is then passed through several steps of pre-processing. First the signal sample streams are segmented into blocks for two reasons: (1) the signal is segmented for following block-based statistical processing; and (2) more importantly, the block size is adjusted according to the input signal power variation and instant SNR estimation. Currently a proportional mapping is applied for the configuration:

$$N = N_o / (1 + SNR/K) \qquad (3)$$

Here N_o is the initial block size in the number of samples, *SNR* is in dB, and *K* is the *SNR* resolution in dB. Normally, *K* is set to be between 2 and 5. *K* can be set to a very large value to make *N* constant. *N* is also upper-bounded by a limit that depends on the signal envelope variation, the limit is set when the envelope variation exceeds a threshold value in dB. The threshold can be reconfigured to fit different channels' slow fading characteristics. For narrowband public safety modulation signals, some empirical values are set for AWGN channels. Under AWGN line of sight channel condition, the threshold is set to be 0 ~ 1dB; in an in-door lab environment without line of sight, the threshold is set to be 0 ~ 3dB. Such a signal segmentation scheme is adaptive to both instant signal SNR and propagation fading, which greatly enhances the stability of statistics calculated from each signal sample block.

For the rest of the pre-processing, the signal samples in each block are centered and scaled. Centering is to remove possible DC offset from radio front-end.

$$S_C = S_i - \frac{1}{N}\sum_{i=1}^{N} S_i \qquad (4)$$

Signal scaling is to normalize the signal 's amplitude to a common full-scale swing range so that the feature extracted at later stages are less sensitive to instantaneous power variations; this is similar to Automatic Gain Control (AGC) in a conventional receiver [42]. There are two reference values commonly used for normalization, mean or variance. To reduce the noise effects, mean based normalization is applied and has been proved by field testing to outperform variance based methods:

$$S_{CN} = S_{C,i} \Bigg/ \left(\frac{1}{N} \sum_{i=1}^{N} \left| S_{C,i} \right| \right) \tag{5}$$

Where S_{CN} is the normalized signal with a swing range of [-1, 1].

After pre-processing, the centered and normalized signal samples are passed to complex envelope processing at the following stage. The following two sections provide the functional design of two classification stages. More details and a comprehensive reference can be found in the resulting publications [21, 29].

3.6 Stage 1: Modulation Identification before Carrier Synchronization

As stated before, although the signal at the output of the pre-processing block is already centered around DC, in real receiver's hardware issues like Local Oscillator (LO) drift and tuning accuracy prevent the "accurate" carrier estimate that many simulation-based papers assume. Therefore, the signal is at quasi-baseband with a residual frequency offset that varies relatively slowly compared to the modulation signal phase transitions.

Since the complex envelope remains the same when the center frequency varies, it becomes a key information source before phase lock. Both the FFT peak and the analytical phase derivative can provide instant frequency estimation, but the phase derivative is very susceptible to noise. It is also difficult to extract a clean nonlinear part from the real over-the-air signal since it is not synchronized. An example both the differential phase and PSD of a 20k baud BPSK signal with 0.35 pulse shaping roll-off are shown in Figure 3.14. Therefore FFT-based Welch periodogram is preferred to provide carrier estimate.

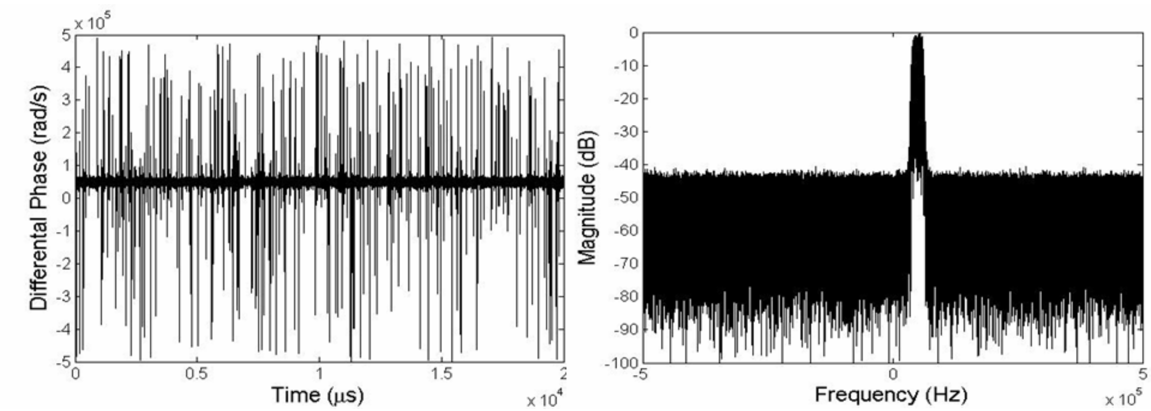


Figure 3.14: Example BPSK signal differential phase and PSD

When the incoming signal only contains real data samples, a Hilbert transform is needed to converted it into complex format. The Hilbert transform is detailed in Section 3.2. Figure 3.15 shows the complex spectrum and features of the BPSK signal from Figure 3.14. The FFT can also be used in generating the Hilbert transform.

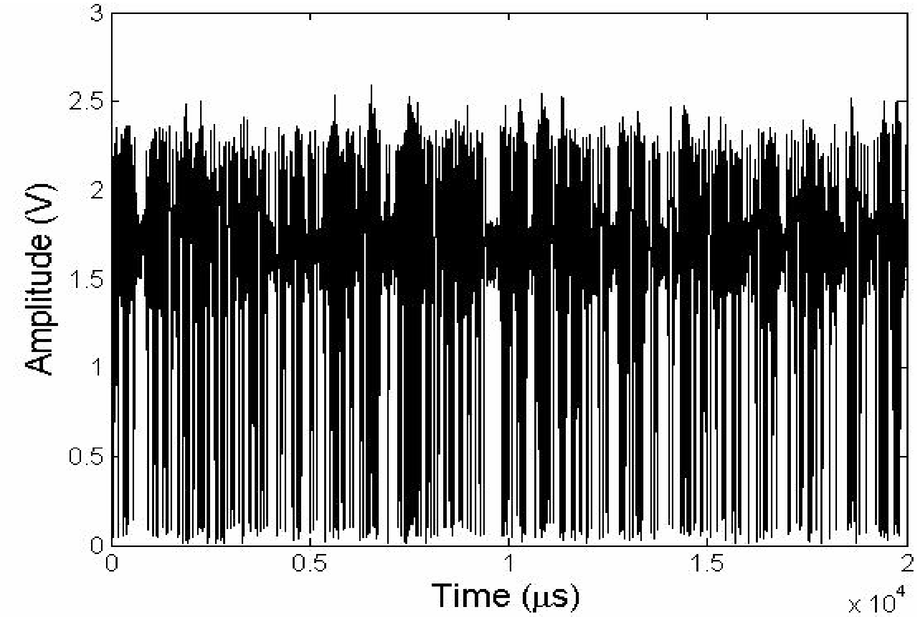


Figure 3.15: Example BPSK signal complex envelope

Among the features that can be extracted from the complex quasi-baseband, the normalized standard deviation of the complex envelope, $\sigma_{CENV} / \mu_{CENV}$, where

$$\sigma_{CENV} = \sqrt{\left(\frac{1}{N}\sum_{n=1}^{N} CENV_n^2\right) - \left(\frac{1}{N}\sum_{n=1}^{N} CENV_n\right)^2} \quad (5)$$

$$\mu_{CENV} = \frac{1}{N}\sum_{n=1}^{N} CENV_n \quad (6)$$

is the most stable and separable feature characterizing modulation signal groups. On the other hand, most complex amplitude and envelope based features, even with high order statistics like kurtosis, are strongly correlated.

Due to adaptive pre-processing, the feature extraction is also based on the block processing adaptive to the input signal SNR. White noise is effectively suppressed by variable block averaging. As shown in Figure 3.16, even with 5 dB SNR, the first-order temporal feature sets from different modulations are still fully separable. In fact, as far as the feature is separable in theory, feature processing in the temporal domain can also get good results at low SNR. The reason is that the complex envelope is not as noise-sensitive as phase; thus envelope-based features perform better than phase statistics.

Although the envelope-based feature is robust and does not require carrier synchronization, it has limits when classifying higher-order modulations (like PSK8, QAM8, QAM16, etc). In the first stage of the signal classification process, it is enough to classify analog modulations and simple digital modulations. Together with keying rate detection, it is also enough to tell the carrier synchronizer to adapt to a specific modulation group - i.e., is it real or quadrature, analog or digital, linear or nonlinear?.

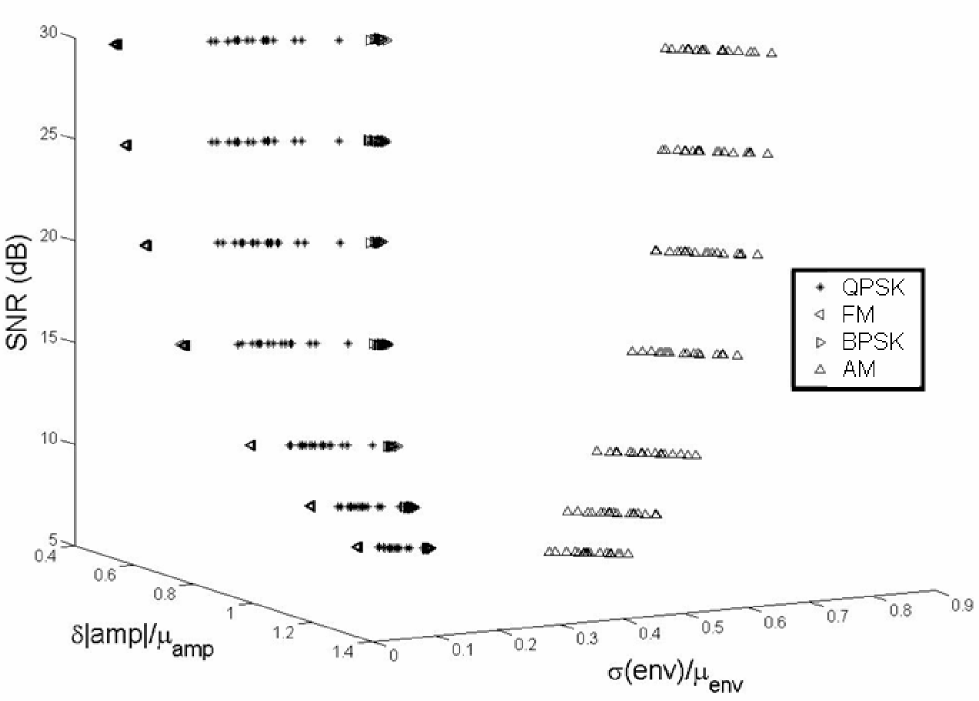


Figure 3.16: Temporal feature set with SNR adaptive feature extraction

Since the features used here have just one or two dimensions, the K-NN feature clustering method is applied. The designed K-NN is a one or two dimensional feature slicer, which contains an adaptive threshold grid that separates different groups of modulations. A K-NN feature slicer classifying four modulation signals is shown in Figure 3.17

As seen from Figure 3.17, the decision boundaries are achieved from mutually exclusive training. Due to the noise adaptive feature extraction, the slicers are trained very easily, taking subseconds to train all four modulations in Matlab and taking almost no time in C++. An example convergence curve for AM training is shown in Figure 3.18.

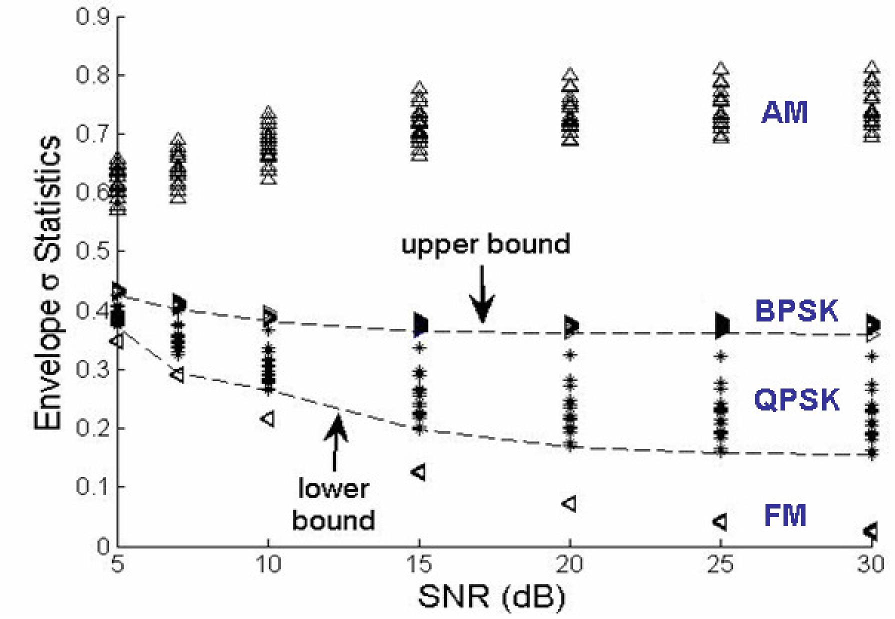


Figure 3.17: K-NN feature clustering with four modulation schemes

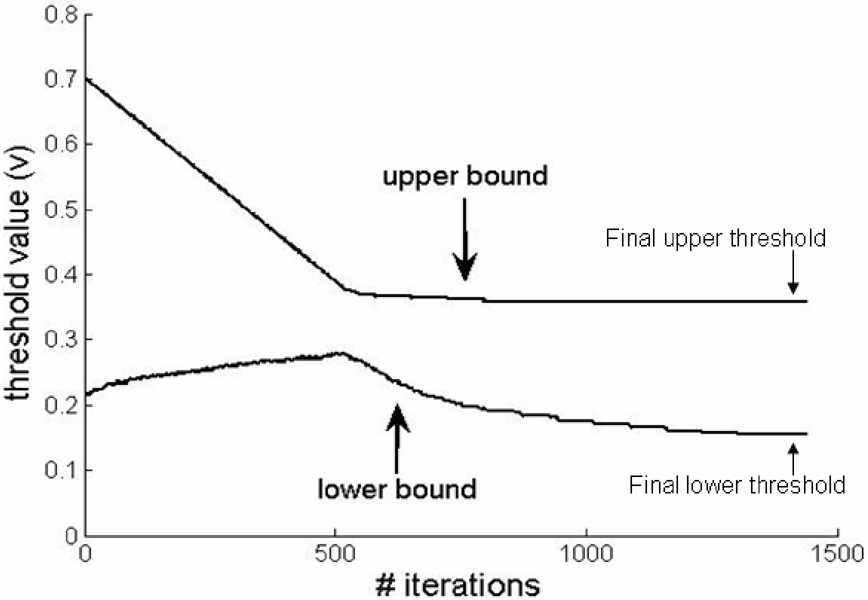


Figure 3.18: Mutual exclusive training curve for BPSK in K-NN slicer

The classification correction rate directly depends on processing gain from the block averaging feature extraction block. The main tradeoff is between processing delay and accuracy for different incoming signal quality. The feature extraction processing is currently configured to carry block-based accumulating averaging; therefore, as collection length extends, the classification result is more accurate. With in-lab collected public safety air waveforms (15~25 kHz bandwidth) with SNR varying between 5 dB and 30 dB, and with a unit block size of 100 symbols, the modulation is always classified correctly within 10 ~ 20 blocks, which is less than 0.1 seconds.

The baud rate may not need to be detected from feature identification. If the coarse modulation grouping information is enough to refer to pre-known waveform knowledge to “look up” the baud rate. This is very useful for practical applications where only a limited set of modulations are to be classified. In the CWT^2 CR node system, the waveform knowledge base is used to determine the particular keying rate with specific modulation schemes identified at the first stage.

3.7 Stage 2: Modulation Identification after Carrier Synchronization

At complex baseband, temporal feature based classification is preferred when signal quality is relatively good, i.e., with moderate noise and slight distortion. As stated before, temporal feature-based modulation classification is fast and easy to implement. However, this feature set is typically sensitive to noise and distortion. The trade-off between feature robustness and computational complexity is the key.

First-order temporal features basically contain the statistics of a signal’s amplitude, frequency, and phase variation. The choice of features is a trade-off between minimizing the number of features to reduce the ANN input size as well as computational complexity and including all necessary features for reliable modulation classification. The increase of feature set size significantly complicates the computation. It also complicates the classifier design and increases training and classification cost in time and computation. For the second stage baseband modulation classification, five features are defined, described below.

(i) The standard deviation of the direct value of the instantaneous amplitude:

$$\sigma_a = \sqrt{\left(\frac{1}{N}\sum_{n=1}^{N} A_n^2\right) - \left(\frac{1}{N}\sum_{n=1}^{N} A_n\right)^2} \tag{7}$$

(ii) The standard deviation of the envelope of the direct value of the instantaneous amplitude, where ENV is a block averaging of the amplitude:

$$\sigma_{ENV} = \sqrt{\left(\frac{1}{N}\sum_{n=1}^{N} ENV_n^2\right) - \left(\frac{1}{N}\sum_{n=1}^{N} ENV_n\right)^2} \tag{8}$$

(iii) The standard deviation of the direct analytical phase value of the signal:

$$\sigma_{\varphi} = \sqrt{\left(\frac{1}{N}\sum_{n=1}^{N} \varphi_n^2\right) - \left(\frac{1}{N}\sum_{n=1}^{N} \varphi_n\right)^2} \tag{9}$$

(iv) The standard deviation of the differential phase of the signal:

$$\sigma_{|\varphi|} = \sqrt{\left(\frac{1}{N}\sum_{n=1}^{N} \Delta\varphi_n^2\right) - \left(\frac{1}{N}\sum_{n=1}^{N} \Delta\varphi_n\right)^2} \tag{10}$$

(v) The standard deviation of the absolute value of the phase change:

$$\sigma_{|\Delta\varphi|} = \sqrt{\left(\frac{1}{N}\sum_{n=1}^{N} |\Delta\varphi|_n^2\right) - \left(\frac{1}{N}\sum_{n=1}^{N} |\Delta\varphi|_n\right)^2} \tag{11}$$

These features only contain the variance of the first-order temporal signal parameters. Since analytical phase information becomes available at complex baseband, it is emphasized in feature extraction because it is more sensitive to different modulation schemes and less sensitive to white noise than instantaneous amplitude and frequency.

A feature space is constructed with the above defined feature parameters. Due to the multi-dimensionality of the feature space, a statistical pattern recognition method is preferred and performs better than simple K-NN feature cluster used at the first classification stage, which is efficient for simpler feature space [50]. For the pattern recognition technique, there are various classification methods (A good review is given by [100].), among which the Artificial Neural Network (ANN) structure is most suitable for signal classification. The ANN's parallel distributed processing structure provides flexibility and reconfigurability for digital implementation. Its connection-based network topology provides arbitrary nonlinear mapping for complicated feature sets. Simple feed-forward calculation at each node (neuron) is fast for on-line processing. ANNs can learn and adapt to complex, time-varying features and have strong fault tolerance due to its statistical nature; thus they are ideal for modulation schemes.

Most previous work on ANN modulation classification uses Multi-Layer Perceptron Networks (MLPNs) that trade network complexity and computational cost for sample set flexibility and performance stability. Modulation schemes are hidden in the received signals as second-order features which need strong nonlinear processing to extract. They can be grouped by their different modulating mechanisms, such as analog and digital modulation or frequency and phase-amplitude modulation, and these groups may be easily separated, as K-NN is designed at the first stage of the classification system. However, similar modulations within the same group may only have subtle differences, and such differences are further corrupted by noise and distortion. All these effects make it difficult to apply a single, universal neural network for all the modulations under any scenario, even by expanding its size or using complicated neuron operations at each node.

Therefore, A One-Class-One-Network ANN (OCON-ANN) structure is constructed, in which one sub-network is created for each modulation type and these networks each output a value, a probability of a match according to the input signal. There is a judgment network, called MAXNET [101] collects the outputs from all these subnets, and the one with the highest output value wins as the modulation type.

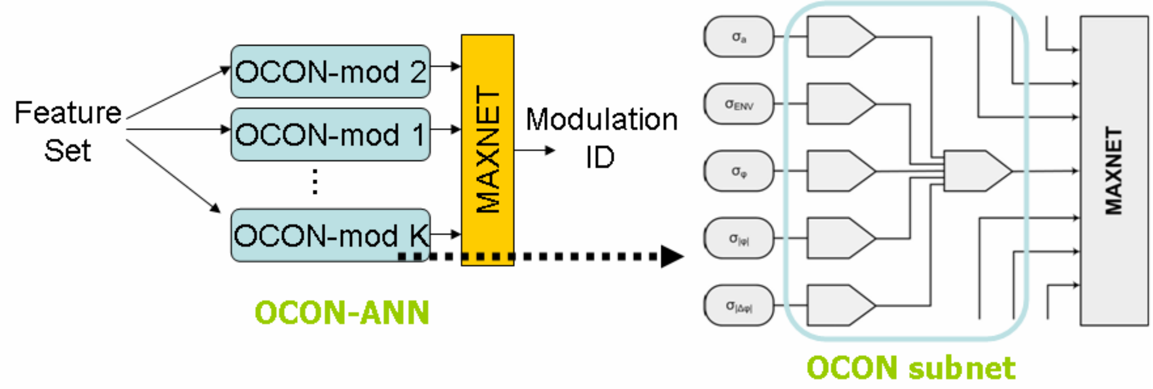


Figure 3.19: OCON-ANN for modulation feature classification

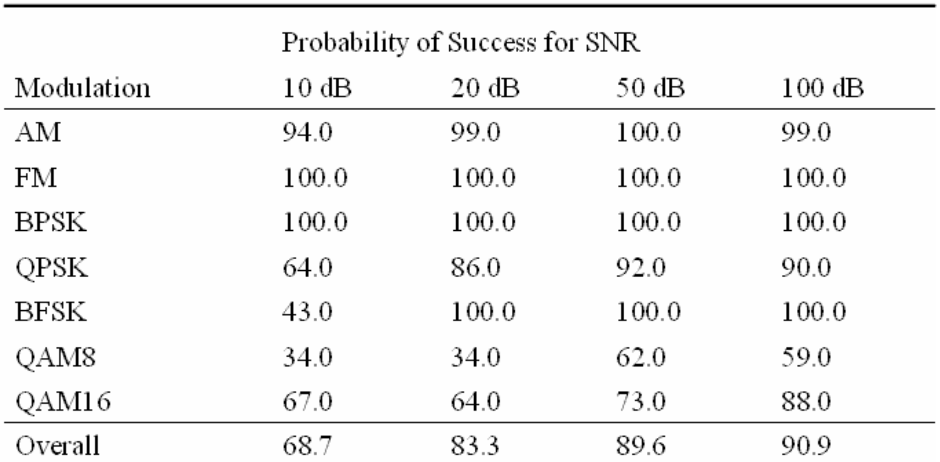
In Figure 3.19 the portion on the left shows a single OCON subnet for one type of modulation. The other incoming lines to the MAXNET are from other OCONs for the other modulation types. Each OCON is a simple MLPN with an input layer that takes five signal features and a single neuron output layer. Each OCON uses a radial basis function for pattern clustering. The right portion shows the topology of the whole network. This system design is simple with small MLPNs and flexible, since adding OCONs for new modulations is trivial. And more importantly, when the modulation signal sample set is changed due to application, the reconfiguration of the classification system only needs to load or remove a certain OCON subnet with its related connection configuration. The whole network need not be retrained. These networks also benefit from SNR adaptive block processing similar to the first classification stage; thus the performance is more stable than most results from previous work in the literature.

A simulation framework is constructed to prototype the second classification stage. The details of the simulation system is given in [29]; some key results and conclusions are provided here.

We created seven modulations: AM, FM, BPSK, BFSK, QPSK, QAM8, and QAM16, which are the typical modulations used in narrowband wireless communications. We evaluated the performance by creating real-world signals that were band limited and the digital signals were pulse shaped. We also investigated the performance dependency on different SNR values in AWGN environment. We used SNR values of 50, 20, 10, and 0 dB as well as a noiseless system (100 dB). A total of 800 signals of each modulation type were created to the above specifications. One hundred of these signals were used to train each OCON and 700 signals were used in the testing of the parallel system.

The simulation result is shown in Table 3.1. It shows that a good neural network design can effectively reduce the feature set without losing performance. Overall probability of successfully classifying any signal is over 80%, which is comparable with the results presented in many of the previous works with significantly larger feature sets and network sizes. Since a compact feature set is important to on-line processing implementation, this OCON design is very promising. However, performance varies significantly with different modulations. Two modulation groups, analog and digital, can be separated ideally with a wide range of SNR. It is noticeable that most of the confusion occurs when trying to differentiate between the higher order quadrature modulation signals, QAM8, and QAM16. The feature space help explain how OCON-ANN works and performance degrades with noise.

Table 3.1: Modulation classification performance using temporal statistics



	Probability of Success for SNR			
Modulation	10 dB	20 dB	50 dB	100 dB
AM	94.0	99.0	100.0	99.0
FM	100.0	100.0	100.0	100.0
BPSK	100.0	100.0	100.0	100.0
QPSK	64.0	86.0	92.0	90.0
BFSK	43.0	100.0	100.0	100.0
QAM8	34.0	34.0	62.0	59.0
QAM16	67.0	64.0	73.0	88.0
Overall	68.7	83.3	89.6	90.9

The feature space constructed with multiple feature parameters is the basis for pattern classification methods to distinguish different patterns represented by those features. And the feature space is illustrated by plotting the subset of features to show how different signals may be classified by the neural network's clustering nature. In the feature space, while certain features group modulation schemes together, other features help separate them. An example feature subspace with standard deviation from phase, envelope and amplitude information is shown in Figure 3.20.

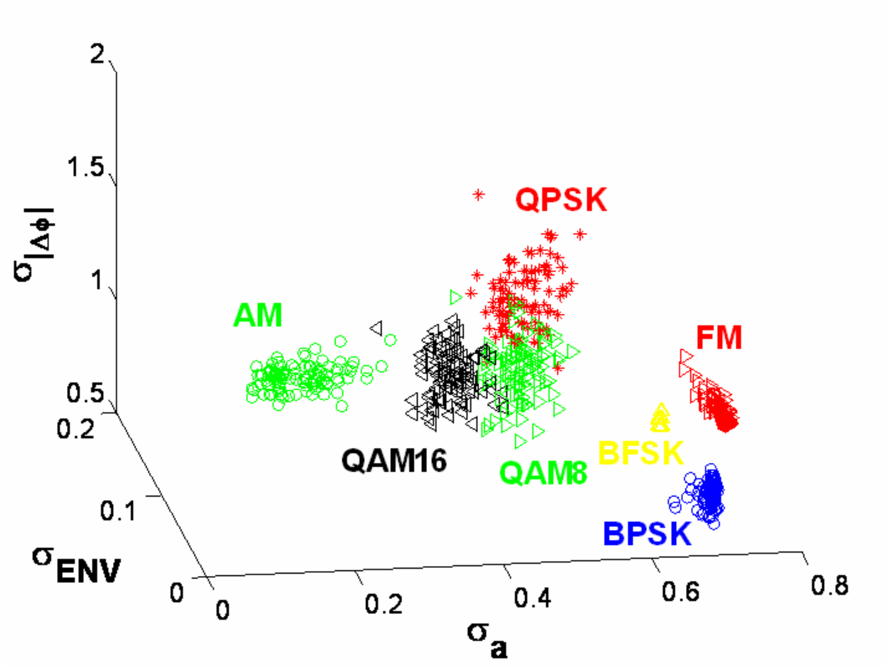


Figure 3.20: Example feature space of temporal waveform statistics, SNR = 40dB

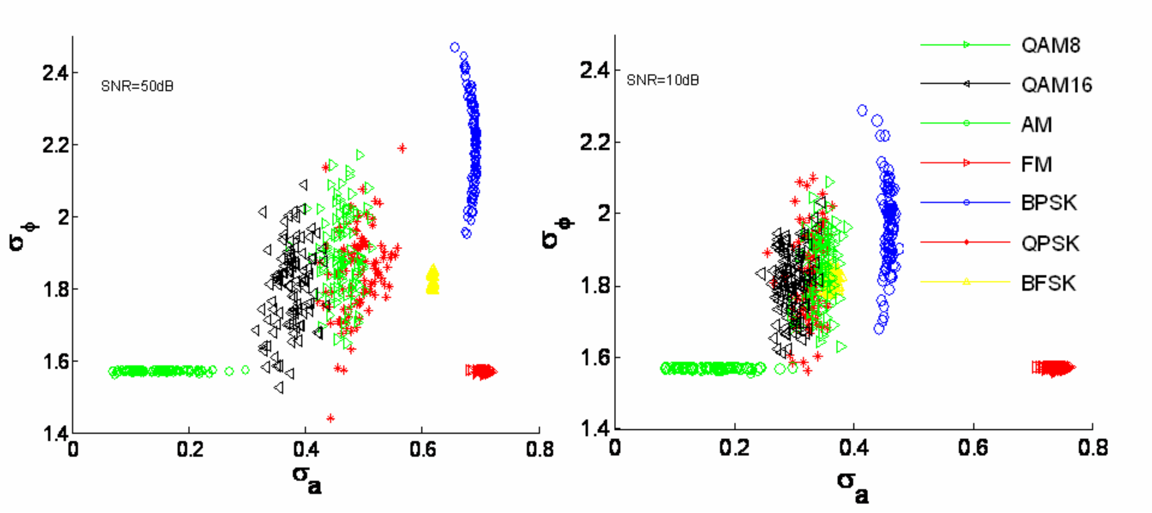


Figure 3.21: 2D temporal feature space with different SNR values

In Figure 3.21, the 50 dB SNR (low-noise) and 10 dB SNR (noisy) results are compared. The separation is reduced by noise. Since the differentiation capability is defined by feature definition, the neural network classifier design is nothing but a trade-off between the network size and nonlinearity.

Adding features equivalently increases the dimensionality of the feature space. And from Figure 3.22 it is clear that the additional dimension helps separate two groups. However, in the time domain, it is always difficult to separate high-order modulations such as QAM8 and QAM16 unless high-order statistical moments are calculated to "extract" their high-order modulation-specific feature. This is the common bottleneck of temporal waveform statistics, which is why vector space and spectral features should be used jointly.

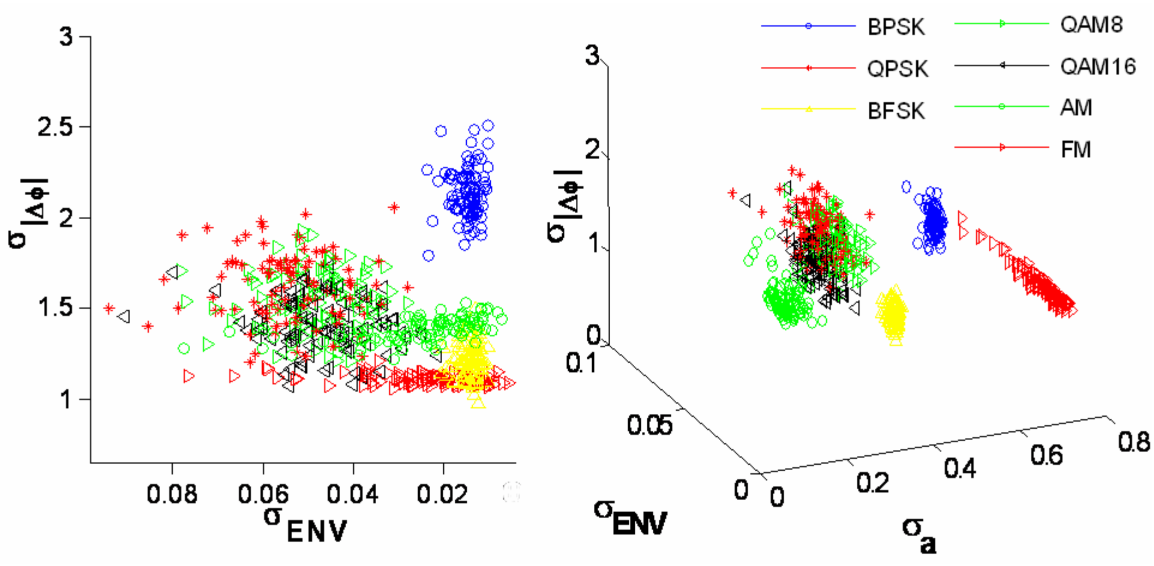


Figure 3.22: 2D and 3D feature space comparison at 10dB SNR

Each OCON is trained to have a focused modulation cluster. A classical back propagation algorithm is used in training. In Matlab, training took about thirty thousand iterations in twenty minutes, within which about 90% of the optimization of the networks was done in the first half of the iterations. One example training convergence curve for BPSK signal is shown in Figure 3.23. In this network, training is performed off-line, and the operational question is the length of time required to classify a newly observed signal. In the MATLAB based simulation system, the average time to classify is 0.01 seconds for feature extraction and 0.025 seconds for classification. The time requirement would be greatly reduced in a live system because it would be executed on a more computationally efficient platform in compiled language.

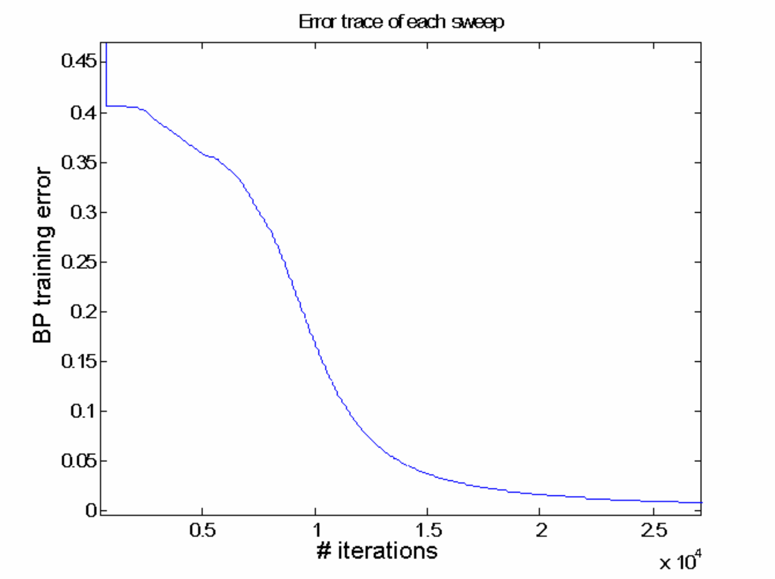


Figure 3.23: OCON subnet training curve for BPSK modulation signal

From these results, it is clear that a better performance is generally achievable by adding memory and computational cost. A good design is only marked by a practical trade-off. The five features currently picked are the empirical results from currently simulation. There may be other temporal features that are more robust against noise. Specifically at the second stage, the constellation pattern is available after symbol timing and can be jointly used to classify high order modulations.

Vector space analysis applies only for digital linear modulations whose complex symbol decision statistic can be represented as constellations in the in-phase/quadrature (I/Q) plane. This provides a graphical insight into the underlying modulation schemes [102]. It is a byproduct of demodulation, which is typically used for accurate phase estimation and fine tuning [103]. After the processing gain through matched filter, it is less sensitive to noise. The constellation feature can be extracted as a complex histogram of the decision statistic at the output of matched filter. The histogram distribution can be easily compared to a known set of modulation schemes. Since currently application doesn't include high order modulations, vector space level signal feature classification module is not implemented yet.

3.8 Between Stage 1 and 2: Synchronization

The heart of a radio receiver is carrier phase lock that makes the complex baseband signal available to feed the symbol timing loop for demodulation. A good tutorial on carrier synchronization design for different modulations is given by [94]. However, as stated before, the carrier synchronization and symbol timing for the modulation classification system should be reconfigurable for different modulations groups, analog or digital, linear or nonlinear, real or quadrature.

Carrier synchronization consists of carrier recovery and carrier phase lock. Modulation information is the key to the configuration of both operations. The principle of carrier recovery is to apply nonlinear energy extraction for distinct frequency components, and synchronization is maintained by feedback of phase error [51, 76]. Typically, carrier recovery depends on one of two assumptions, using a pilot signal or assuming a symmetric baseband spectrum to enable carrier regeneration through nonlinear operations. In the pilot-aided case, the receiver should know the modulation standard to recognize the pilot tone; in the carrier regeneration case, the receiver also has to know the modulation scheme in advance to apply an appropriate nonlinear operation. Carrier phase lock also depends on modulation information. Analog frequency modulation needs a simple PLL with an audio loop bandwidth, while digital frequency modulations may need multiple PLL branches for different tones. For digital linear modulations, the phase error detecting algorithm may need to be adapted to either real or quadrature form, such as BPSK and QPSK modulations. Keying rate also plays an important role in setting the loop bandwidth. As stated earlier, keying rate can be detected via envelope based features, or via knowledge lookup based on other modulation features already identified.

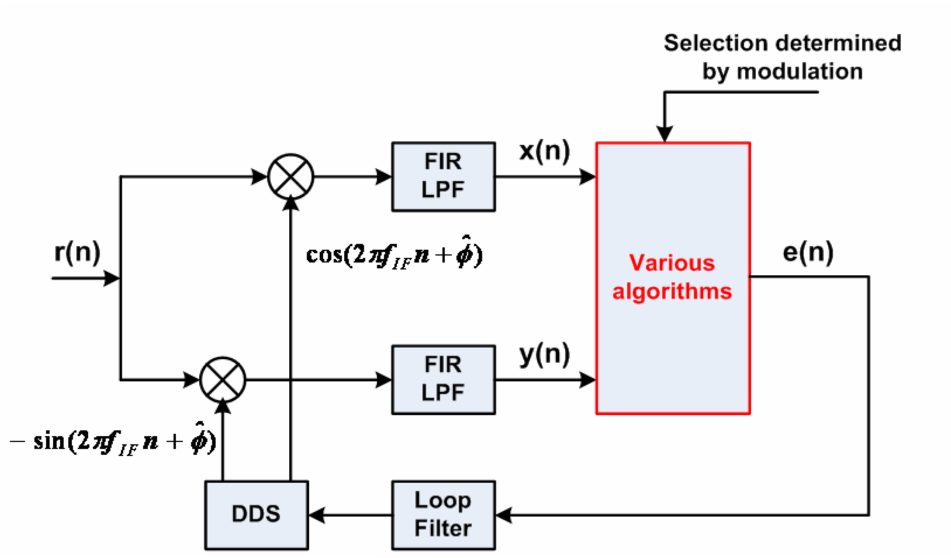

Selection determined by modulation
FIR LPF
x(n)
r(n)
$\cos(2\pi f_{IF} n + \hat{\phi})$
Various algorithms
e(n)
FIR LPF
y(n)
$-\sin(2\pi f_{IF} n + \hat{\phi})$
DDS
Loop Filter

Figure 3.24: Modulation-adaptive carrier synchronization loop

As shown in Figure 3.24, a quadrature multiplication structure [95, 104] is selected because the in-phase and quadrature branches can be flexibly reconfigured for both real and quadrature modulations. With full digital implementation [97], the phase error detector and loop filter are reconfigurable for both linear and nonlinear modulations; the phase error detectors can also be configured as an FM demodulator; and the structure can further be replaced by a low-pass FIR filter for analog AM signals. There are three important features of the implementation of this modulation-general phase lock loop. First, since the input signal is in complex form, the conventional I/Q branch multiplication processing can be simplified to a single complex multiplication. And the following conventional I/Q phasing calculations are simplified as complex analytical phase differentiation. Second, since the input signal is not at a conventional IF but the complex quasi-baseband, the complex "carrier" phase lock loop actually tracks a residual frequency offset that results in a phase varying slower than the phase transitions of the baseband modulation signal. Therefore, the loop filter can be set with a low cutoff frequency, and thus the carrier tracking could be less sensitive to specific modulation orders within the same modulation group. Third, with all the system implemented with complex digital processing, all the timing errors at carrier level or symbol level are represented as complex analytical phase (or angle) sliding in the unit of radians per sample. Thus the carrier phase error can be corrected with feedback from the symbol timing block. In this way, a big tracking loop is formed, combining the carrier synchronization and symbol timing, to correct the overall timing by correcting the overall phase error, as shown in Figure 3.25.

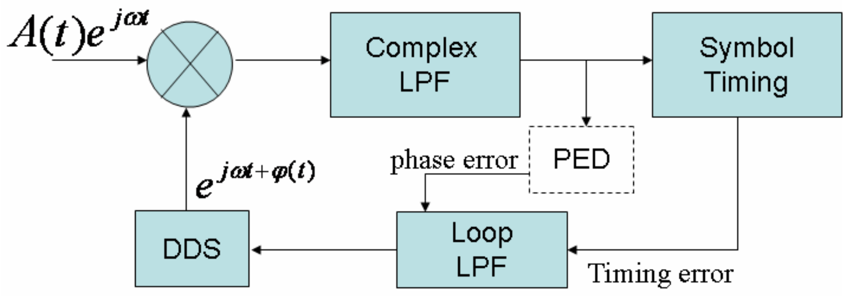


Figure 3.25: Complex quasi-baseband digital synchronization implementation diagram.

It has been proved in the CWT^2 CR system field testing that such a generic carrier synchronization structure with second-order carrier recovery can synchronize both AM and BPSK signals. And the general synchronization loop works well for linear digital modulations and does not need modulation specific phase error detection. This makes it suitable for the adaptive two-stage signal classification system.

Symbol timing is essential for coherent demodulation. Although various symbol synchronization and timing algorithms are available [105], most methods are designed for specific signal parameters like modulation, symbol rate, pulse shaping filter, etc. To maximize generality, an early-late gate loop (also called Mueller-Muller algorithm [106]) is preferred for the symbol timing block to work with the classification system. The early-late gate timing loop depends neither on the specific modulation scheme nor pulse shape. It only assumes that the pulse is symmetric, which is almost always valid for digital modulation signals. With digital multi-rate processing implementation, the noninteger ratio between sample rate and symbol rate enables a theoretically infinite timing resolution due to the sliding effect.

Both the adaptive carrier synchronization and early-late gate symbol timing blocks are implemented in CWT^2 software radio system called the waveform framework (See Chapter 6).

3.9 Signal Classification Testbed Integration and Demonstrations

3.9.1 Testbed 1: Signal Classification System for NIJ Demo 2005

The first signal classification testbed system is implemented fully in Matlab. This testbed served as the demonstration system for NIJ CommTech project [18] Program Review in 2005. The layout of the demo system is shown in Figure 3.26.

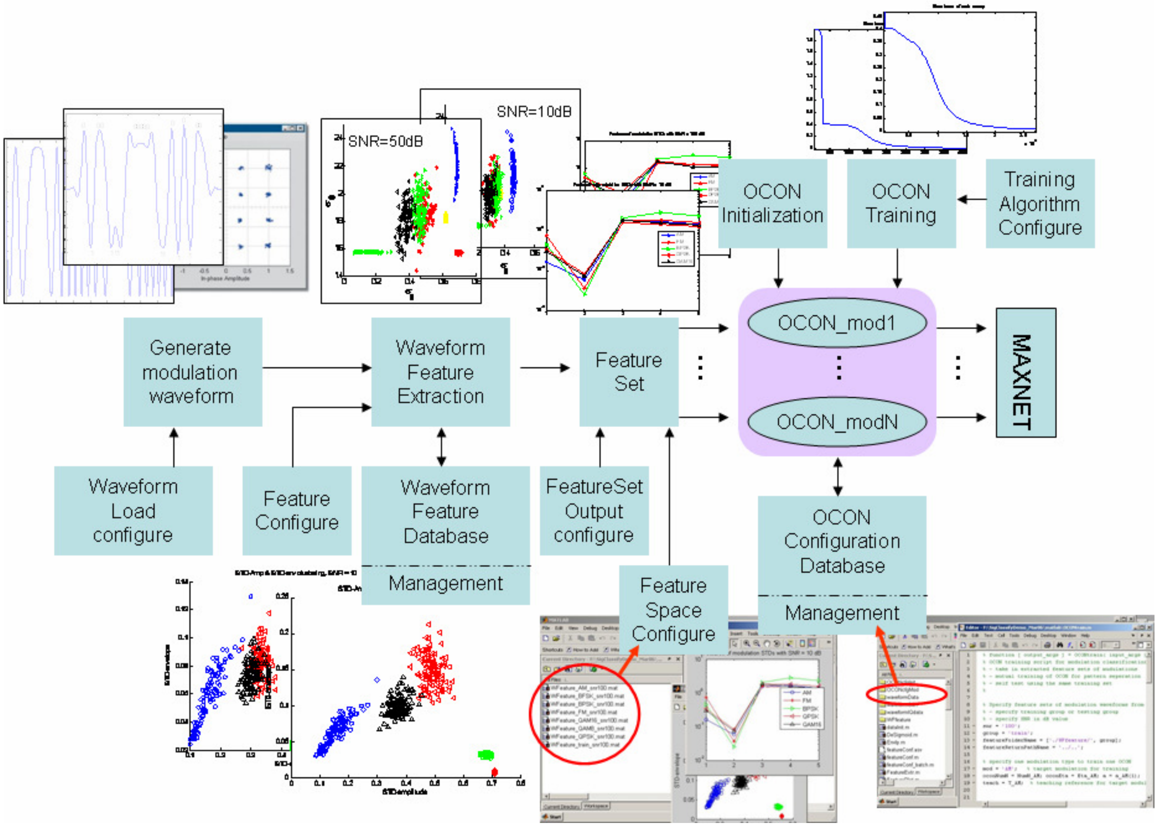


Figure 3.26: Signal classification NIJ demo 05 system layout

This testbed is constructed to verify and benchmark the baseband modulation classification capability based on OCON-ANN approach. The theory behind the system design and the simulation results are already explained in Section 3.7 and are also detailed in [29]. Note that the classification system follows an open-structure, modular design approach in building the OCON meta-network system. A database is created for storing the key knowledge of the classification system, including feature set definition, extraction methods, OCON subnets configuration, and network training configuration. The whole system can be flexibly reconfigured for different modulation schemes, different signal quality, and performance requirements, i.e., using a longer training for a better classification result.

3.9.2 Testbed 2: Smart Receiver Anritsu Demo for SDRF 2006

After basic theory and system functionality were verified and implemented with the first testbed. A fully functional "smart receiver" system was designed and implemented with GNU radio USRP hardware platform and Anritsu Signature signal analyzer. This smart receiver system is able to detect the signal over the air, recognize the waveform format including modulation and frame format, and then use the recognized waveform knowledge to instantly configure a software-defined signal processing chain to receive the target waveform and provide the data or voice service. Together with the waveform recognition system, a fully reconfigurable software defined radio platform has also been developed to provide flexible waveform generation.

The smart receiver system is the integration of the waveform recognition and software radio platform. Both the system's waveform recognition and reconfigurable waveform generation are demonstrated at Software Defined Radio Forum Technical Conference in November 2006. The demo system layout is shown in Figure 3.27.

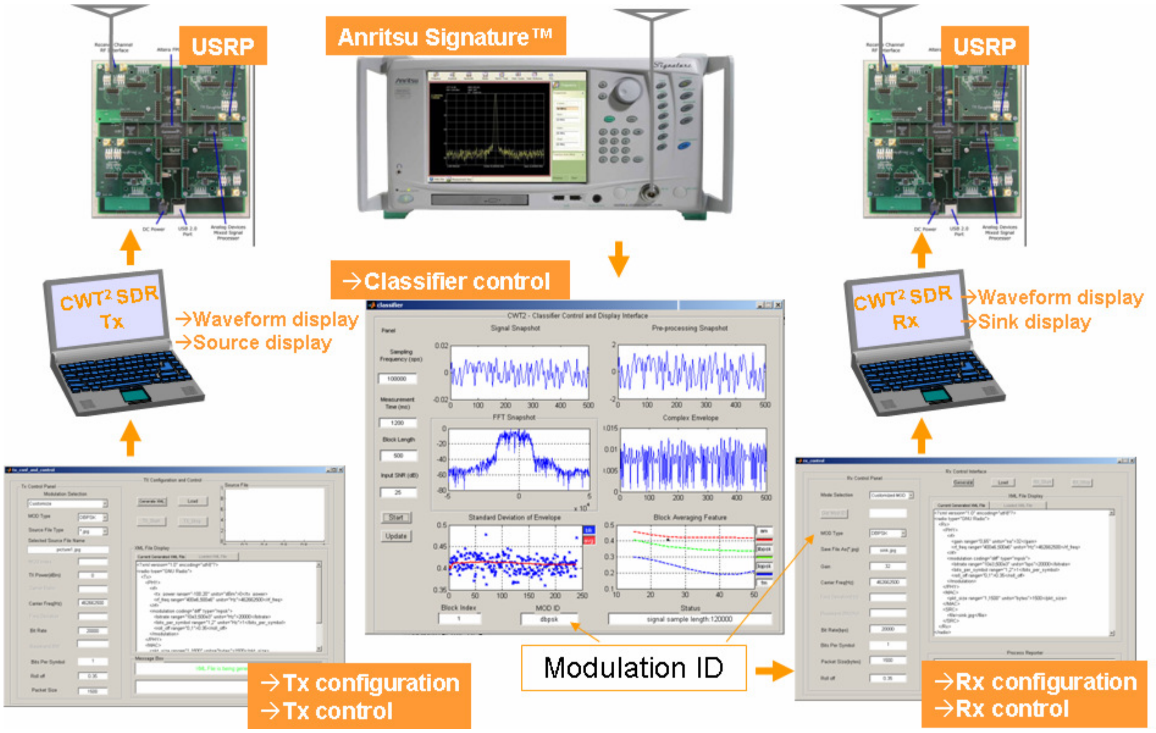


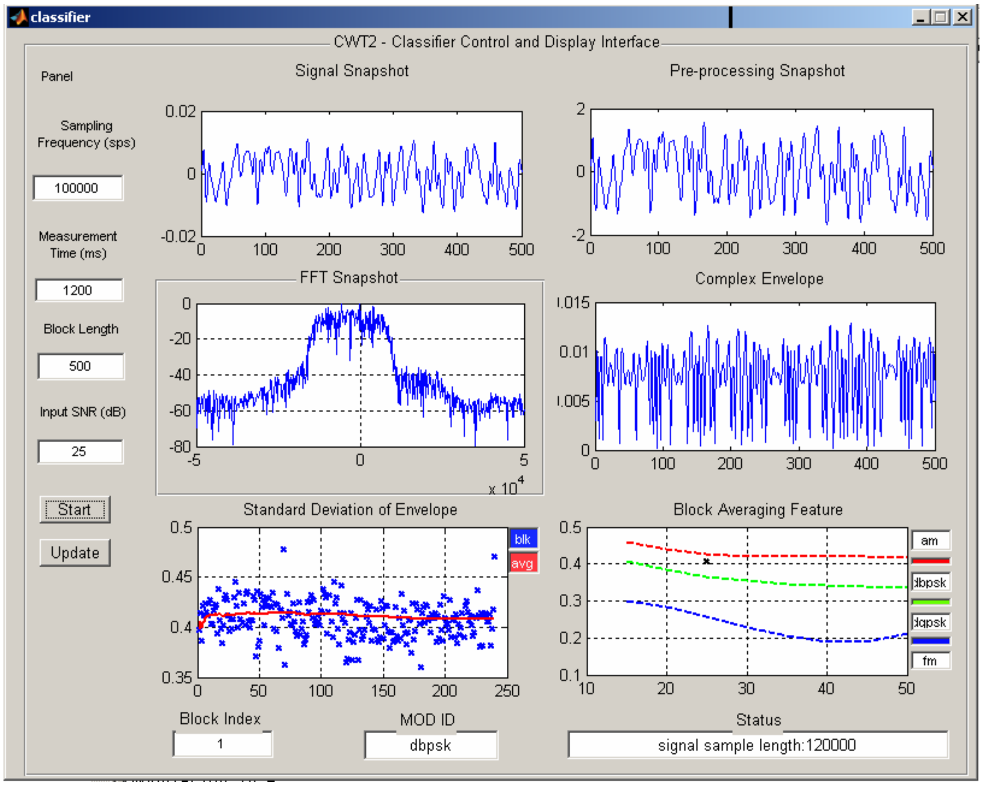
Figure 3.27: CWT^2 smart receiver SDR’06 demo system layout

Specifically, the demo system comprises three major functional blocks. The first block is the waveform recognition subsystem. It includes three set of algorithms to accomplish the recognition of a complete waveform. The first algorithm controls the radio front-end to detect the energy in the air and collect sensed signal samples (See Section 3.2.); the second algorithm set carries the complete waveform recognition procedure from RF to packet format (See Sections 3.3 through 3.8.). The frame format and link protocol are recognized through waveform knowledge lookup after the signal PHY parameters are classified. The third algorithm set manages waveform and link protocol knowledge according to the recognized radio environment and generate a standard configuration and control interface to the radio platform.

The second functional block is the software radio platform. It is developed using the GNU Radio software library and serves to generate full waveform stack from PHY to NET layer in software. In this platform signal flow is implemented fully in complex digital processing, especially the synchronization and modem design (see Section 3.8).

Four groups of modulations were supported at that time, AM, FM, BPSK and QPSK, for all of which the PHY and MAC parameters are fully reconfigurable. Both voice and data services are supported. For digital link the data rate can be adjusted from 20kbps to 350kbps seamlessly. Waveform reconfiguration and link mode can be switched in less than 0.1 second, which is promising for real-time applications.

The third functional block is the Graphical User Interface (GUI). Three GUI displays were developed, specifically for the control and function illustration of waveform transmit, waveform receive, and waveform recognition. Taking a closer look at waveform recognition GUI in Figure 3.28, we see that there are several real-time snapshots of the detected signal, some example features and its modulation classification criterion. The input signal SNR is estimated and block processing can be automatically adapted or manually controlled at the left side bar. The bottom gives some status information and modulation identification result. Note that the incoming signal is at complex quasi-baseband with frequency offset PSD (See Sections 3.5 and section 3.8.) shown on the middle left plot. The plot on the bottom left illustrate how adaptive block averaging help stabilize modulation feature set (See Section 3.5.). The plot on the bottom right is the K-NN decision grid used to classify the four implemented modulation groups (See Section 3.6.).



Sampling Frequency (sps): 100000

Measurement Time (ms): 1200

Block Length: 500

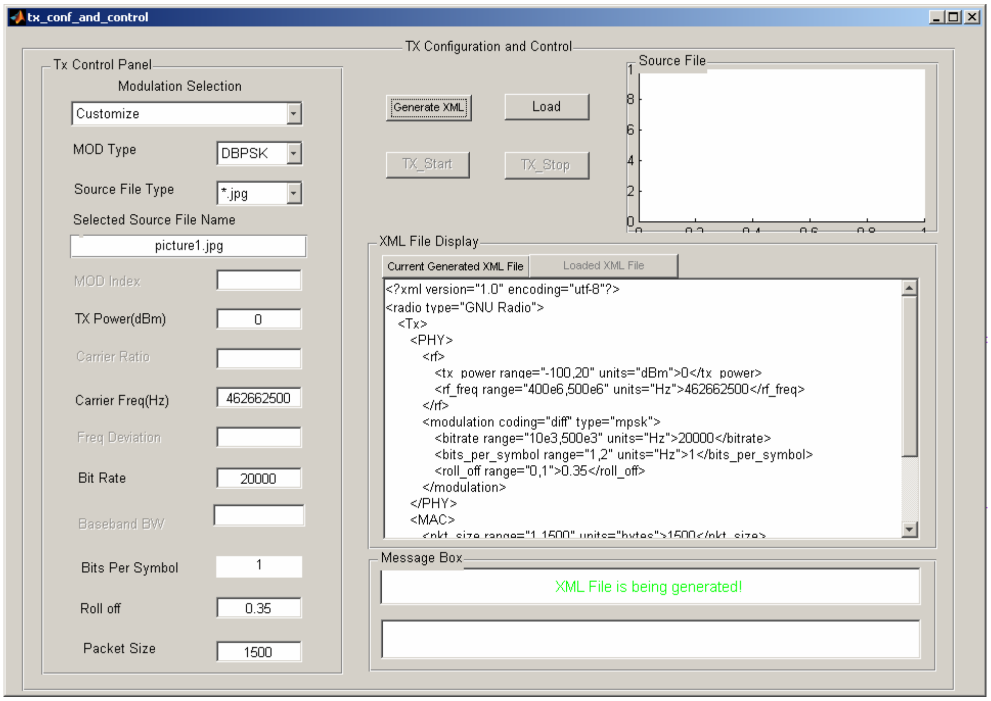
Input SNR (dB): 25

Block Index: 1

MOD ID: dbpsk

Status: signal sample length:120000

Figure 3.28: CWT^2 smart receiver demo signal classifier GUI



Modulation Selection: Customize

MOD Type: DBPSK

Source File Type: *.jpg

Selected Source File Name: picture1.jpg

MOD Index:

TX Power(dBm): 0

Carrier Ratio:

Carrier Freq(Hz): 462662500

Freq Deviation:

Bit Rate: 20000

Baseband BW:

Bits Per Symbol: 1

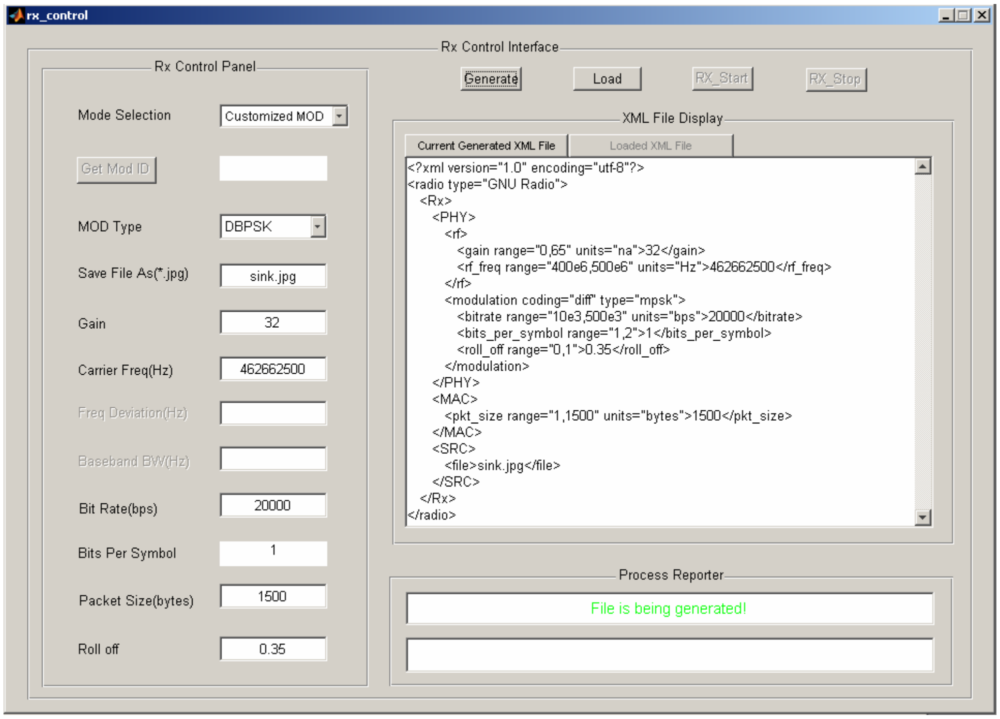
Roll off: 0.35

Packet Size: 1500

```
<?xml version="1.0" encoding="utf-8"?>
<radio type="GNU Radio">
  <Tx>
    <PHY>
      <rf>
        <tx_power range="-100,20" units="dBm">0</tx_power>
        <rf_freq range="400e6,500e6" units="Hz">462662500</rf_freq>
      </rf>
      <modulation coding="diff" type="mpsk">
        <bitrate range="10e3,500e3" units="Hz">20000</bitrate>
        <bits_per_symbol range="1,2" units="Hz">1</bits_per_symbol>
        <roll_off range="0,1">0.35</roll_off>
      </modulation>
    </PHY>
    <MAC>
      <pkt_size range="1,1500" units="bytes">1500</pkt_size>
```

XML File is being generated!

Figure 3.29: CWT^2 smart receiver demo waveform transmit GUI



Mode Selection: Customized MOD
MOD Type: DBPSK
Save File As(*.jpg): sink.jpg
Gain: 32
Carrier Freq(Hz): 462662500
Freq Deviation(Hz):
Baseband BW(Hz):
Bit Rate(bps): 20000
Bits Per Symbol: 1
Packet Size(bytes): 1500
Roll off: 0.35

```
<?xml version="1.0" encoding="utf-8"?>
<radio type="GNU Radio">
  <Rx>
    <PHY>
      <rf>
        <gain range="0,65" units="na">32</gain>
        <rf_freq range="400e6,500e6" units="Hz">462662500</rf_freq>
      </rf>
      <modulation coding="diff" type="mpsk">
        <bitrate range="10e3,500e3" units="bps">20000</bitrate>
        <bits_per_symbol range="1,2">1</bits_per_symbol>
        <roll_off range="0,1">0.35</roll_off>
      </modulation>
    </PHY>
    <MAC>
      <pkt_size range="1,1500" units="bytes">1500</pkt_size>
    </MAC>
    <SRC>
      <file>sink.jpg</file>
    </SRC>
  </Rx>
</radio>
```

File is being generated!

Figure 3.30: CWT^2 smart receiver demo waveform receive GUI

It is also useful to take a look at both transmit and receive GUIs in Figure 3.29 and Figure 3.30 respectively. On both GUIs, the left bar provides all reconfigurable waveform parameters for a chosen modulation group. A standard waveform description is automatically generated in XML, which can be loaded to the software radio platform via a platform independent interface.

The signal classification and GUI systems are coded in Matlab. Their control interface is implemented using standard TCP/IP sockets; thus their functionalities can be network distributed. As shown in Figure 3.28, for the demo system, the transmit GUI is with one SDR node implemented in Linux-GPP hardware; the receive GUI is with another SDR node. The signal classification GUI controls the classifier subsystem using Anritsu Signature Spectrum analyzer as the radio front-end, while both the transmitter and receiver nodes are using USRP radio front-end. They can also be fully integrated as a one-node smart receiver shown in Figure 3.31, where all the Matlab codes are now converted to full C++ object oriented real-time implementation. The development of this demo system provides a lot of theoretical and engineering experiences from algorithm coding level up to system integration level. It also serves as a starting point of the development of a fully functional cognitive radio node system, which is detailed in Chapter 7.

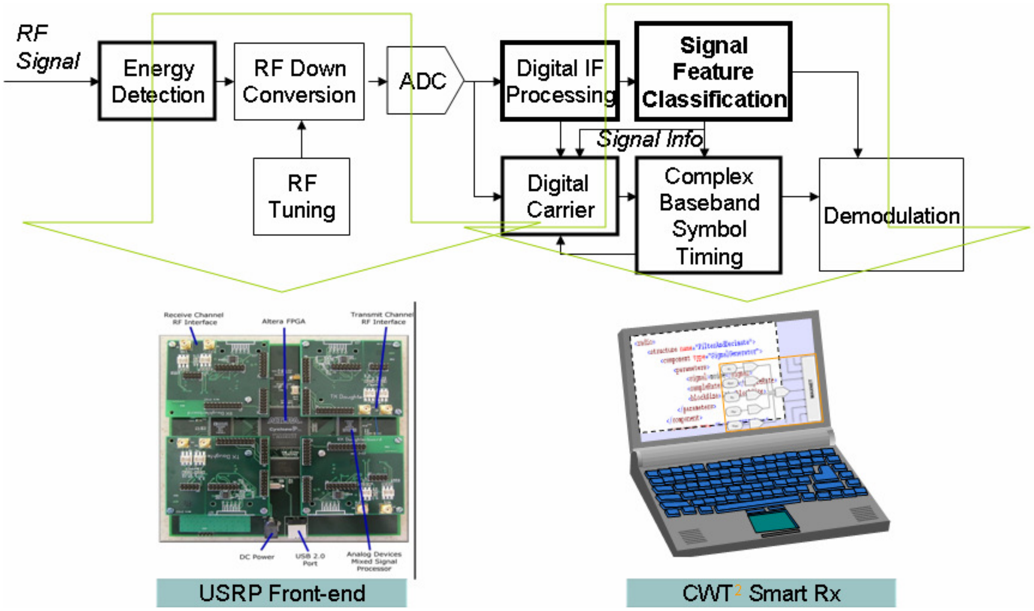


Figure 3.31: CWT^2 integrated smart receiver node system

3.10 Summary

Chapter 3 presents a systematic design to achieve radio environment awareness. In the CWT^2 CR solution, the radio environment is defined to contain information about existing waveforms on the spectrum and their propagation characteristics. According to our target public safety application needs, the waveform recognition becomes the research focus. First, a spectrum scan is needed to identify existing energy components. To achieve a good trade-off between measurement reliability and computational cost, a block-based hierarchical FFT algorithm is designed to sweep the segmented frequency portions with Welch periodogram averaging. Then the signal at the channel of interest is extracted, and its parameters are identified by a signal classification system. The signal classification system has two stages before and after carrier synchronization. The processing at the later stage is adaptive to the results from the previous stage to extend classification capability. Feature pattern recognition approach is applied to achieve both the generality and efficiency in classification. Classic pattern recognition techniques are modified to meet communication signal features. Both the spectrum scan and signal classification are configured adaptive to incoming signal quality such as SNR and envelope fading characteristics, thus the performance stability is greatly improved. A modulation-adaptive carrier synchronization and symbol timing subsystem is designed to aid the signal classification system. The carrier synchronization loop can be configured to either real PLL for nonlinear modulations, or quadrature multiplication structure for linear modulations. The symbol timing loop uses early-late gate timing algorithm that is applicable to general pulse shaped symbols.

During the research progress, two simulation testbeds are developed. The first testbed is a full-Matlab signal classification system focusing on baseband modulations. The second testbed is a smart receiver system implemented in both Matlab and C++. Its signal classification focuses on un-synchronized signal modulations. It also includes a developing version of a SDR platform which provides all the testing waveforms and link capabilities.

Chapter 4: Radio Domain Reasoning and Learning

4.1 Machine Learning Principles and Algorithms

Machine learning has been well-documented with both criticisms [107] as well as successes [108], especially in narrowly defined, well-bounded applications. While machine learning in wireless communications is still a bounded problem, the technical demands for multi-band, multi-mode, user defined wireless service applications is exceeding today's reconfiguration and adaptation methods that are largely based on fixed rules. Much of the power of a cognitive radio comes from the vision that machine learning capability enables the radio to understand its environment including user needs and optimize its performance by the learning from its field experience.

In the CWT^2 Cognitive radio solution, the basic machine learning structure follows the reinforcement learning principle [39], which is shown in Figure 4.1.

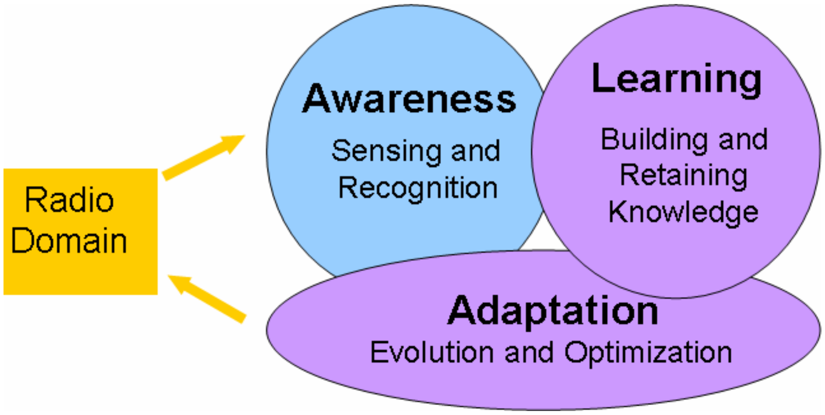


Figure 4.1: Reinforcement machine learning structure

Generally speaking, reinforcement learning builds up intelligence by evolving its knowledge with the experience of the practice instructed by previous available knowledge. In reinforcement learning, appropriate problem-solution association is explicitly formulated as part of the knowledge, and the performance of the practice associated to the problem is evaluated in relative metrics – there's no "optimum" solution, only a better solution or the best solution ever achieved and remembered. And "optimum" is always the goal to achieve by continuing to improve the current solution. Further, reinforced learning emphasizes on-line performance improvement, which involves finding a balance between exploration (of untouched solution space) and exploitation (of current knowledge). The exploration vs. exploitation trade-off in reinforcement learning provides important flexibility between creativity and rationality in learning algorithm design, especially for specific application requirements.

Past experience should help in providing a better solution for the current and future problems. It can bypass a lengthy and complicated analysis and optimization process conducted from scratch. For CR application specific machine learning design, the reinforcement learning principle is applied to constructing intelligence from three major knowledge sources [26]:

(i) Learning by remembering a scenario pattern that includes problem scenario, associated solution and the lessons learned from the solution practice;

(ii) Learning by applying fundamental domain principle to evolve the currently available solution to try for a better performance in a familiar or novel scenario, and gain lessons from practice;

(iii) Learning by applying fundamental domain principles to create a new solution to try for a valid or better performance in a novel or unknown scenario, and gain lessons from practice.

The reinforcement learning principle is the basis for Case Based Reasoning (CBR) [109, 110] and solution making theory [109]. CBR emphasize on a knowledge based reasoning approach in which the solution making relies on the past experience, which is suitable for learning with (1) and (2) knowledge sources listed above. Although CBR is very effective and efficient with familiar problem scenarios, it is well known that when a novel or completely new situation occurs, almost all knowledge-based reasoning and learning mechanisms perform poorly [26]. This falls into the type (3) learning case above, where the association between the encountered problem and previous experience is difficult to generate. Thus a more “creative” solution making mechanism is needed for the problem space that is not well understood. An evolutionary searching algorithm serves as the best candidate for this type of problem [111-114]. Specifically, a Genetic Algorithm (GA) is used to provide creative solutions. With the emphasis on flexible problem space parameterization (chromosome) and performance objective encoding (fitness functions), GA can effectively provide multi-objective solution search on a complicated, unfamiliar problem space [112, 115].

In the CWT^2 CR node system, the machine learning core is the combination of CBR and GA [116]. The balance between rationality and creativity in the reinforcement learning is realized by the cooperation between pattern match based and evolutionary search based solution making.

4.2 Case Based Solution Making and Related Learning

Central to the cognitive radio are notions of reasoning and learning. Reasoning is considered to be the immediate decision process that chooses an action or set of actions using the awareness of current state of the world and available historical knowledge. Learning, on the other hand, is a loop process of accumulating knowledge based on the lesson learned from the experience of past practice.

Case Based Reasoning (CBR), broadly construed, is the process of making solutions to solving new problems based on the solutions of to similar past problems. CBR is within the scope of goal-driven, resource limited machine learning system [30]. Thus it is suitable for radio domain cognition for the following reasons: (1) resources are bounded in radio domain - For example, consider radio platform processing capability, power limitations, etc; (2) the radio domain problem space is also bounded, such as wireless waveform principles, spectrum regulations, user service limitations, etc; and (3) the performance optimization is highly goal driven and governed by application specific objectives.

Compared to other knowledge based reasoning mechanism, CBR has several advantages:

(1) In CBR, there's no need to decompose or generalize the domain knowledge into rules. Problem domain parameters are natural for case representation. And for the solution part the cases can explicitly record problem-solving procedures. CBR emphasizes problem space and solution space collection but simplifies reasoning processing. This is especially useful for efficient solution generation under complicated problem space where general rules may be difficult to elicit.

(2) CBR offers high reasoning efficiency. The solution is made by reusing prior successful solutions rather than repeating the reasoning effort. In knowledge evolving, CBR saves failed solutions as well as successes, and this is useful to avoid repeated mistakes.

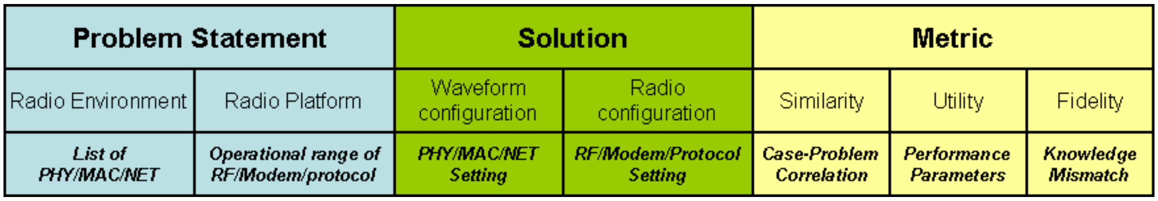
(3) CBR reasoning capability keeps increasing as case knowledge grows. The intelligence is evolved by simple cases growing without new rule/model generation as inevitable in rule-/model based knowledge system. Only a limited set of initial cases are needed at the starting point and they are to be updated or augmented with new cases through practice.

(4) CBR can offer relatively better solutions than conventional rule or model based knowledge systems when facing new situations. When the problem scenario is novel or not well understood, the solutions offered by cases may be more accurate than those suggested by chains of rules, because neither cases nor rules are perfect under this condition. But CBR makes a decision on cases reflecting real problem domain, not "unconfident" rules.

Reference [110] provides a comprehensive survey of CBR research, and [117] serves as a tutorial of CBR theory as well as a complementary CBR survey in Europe. CBR starts with a set of initial cases of standard problem conditions or examples, and then it forms generalizations of these initial cases by identifying commonalities between a remembered case and the target problem and updates the case library with the performance feedback. Such case matching approach in CBR makes it more instructive for providing a known solution for the problem than analyzing it. Therefore it is an effective solution making system for a complicated problem domain for which it is difficult to generate general model or rules.

For the CBR design in the CWT^2 CR learning core, a case library is designed and implemented as a part of overall cognitive radio knowledge base. Each case is a data entry consisting of three parts, the problem statement, the solution to take in response to the problem, and a set of metrics of this solution applied for the specific problem statement. The metric part contains three major parameters, utility, similarity and fidelity. The utility is the performance evaluation of the solution applied to that specific remembered problem. When the cognitive radio sees a new problem, it calculates the similarity of the recognized problem to the problems remembered in its case. The solution of the case that jointly maximizes the utility and similarity is selected as the most applicable one to be deployed. The joint maximization, which could be as simple as a similarity-weighted utility, is preferred because the utility of a solution in a case that is highly similar to the current problem might be less than the utility of a case less similar. The higher utility might outweigh the similarity.

In the case entry definition, each part of a case entry has several data fields, each of which has a set of parameters. Specifically, for the CBR in CR learning core, the case entry definition is shown in Figure 4.2.



Problem Statement		Solution		Metric		
Radio Environment	Radio Platform	Waveform configuration	Radio configuration	Similarity	Utility	Fidelity
List of PHY/MAC/NET	***Operational range of RF/Modem/protocol***	***PHY/MAC/NET Setting***	***RF/Modem/Protocol Setting***	***Case-Problem Correlation***	***Performance Parameters***	***Knowledge Mismatch***

Figure 4.2: Case entry definition of CBR case library in CWT^2 CR learning core

The problem statement part consists of both radio environment and radio platform information. Radio platform information can be pre-loaded and relatively static; it defines the available processing resources of the radio and their specific operation ranges, such as frequency tuning range, power range, supported waveform modes, protocol modes, etc. The radio environment information is obtained from radio domain recognition (See Chapter 3.), which contains a list of waveform and network parameters recognized from the radio environment. The solution part currently consists of a specific waveform and link setting field, and the corresponding radio platform setting field to deploy this link. The metric part contains three fields. The similarity field contains the degree of similarity expressed as a Euclidian distance calculation as explained above. The utility fields can contain many performance evaluation parameters, such as Bit Error Rate (BER), packet delay and transmit power, purely depending on application requirements.

Once the solution is made, it is deployed for practice, and its performance in the real world is evaluated and recorded as the solution's practical utility for the specific problem statement. Also, this practice evaluation can be used to compare the performance anticipation, so that another metric field, the fidelity, can be added to this case entry. Fidelity describes the "confidence" of the performance the solution claims. A general CBR solution making and learning loop is shown in Figure 4.3. Its comprehensive explanation is given in my recent publication [116].

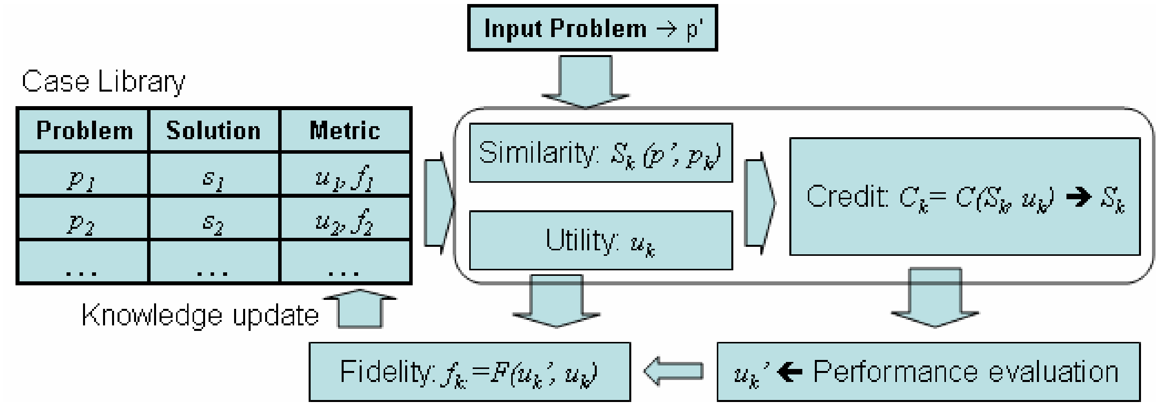


Figure 4.3: Case base solution making and learning loop block diagram

Generally, similarity calculation is a complicated issue. The comparison of similarity between cases is specific to the problem representation and problem space modality. Many strategies are adopted for various applications in CBR research community; a survey is provided in [110].

In the current CR system implementation for the public safety interoperability application, the problem space features standard radio environment scenarios and regular radio operation modes. Therefore, the solution making for standard operations is emphasized in CBR reasoning at the CR learning core, similarity-only case retrieval strategy is used to improve solution making efficiency for on-line processing.

The similarity value is calculated between the input problem and the remembered problems across stored cases. Both the statements of input and remembered problems share the same parametric format so that a simple correlation is used to calculate the Euclidian distance between these two vectors. And the solution from the case with a minimal Euclidian distance from the input problem is picked to be deployed.

By using single value similarity, case retrieval by maximizing similarity is actually a single-objective optimization process. When utility is also applied in the comparison between cases, the case retrieval becomes a multi-objective optimization problem. It would be interesting to see how evolutionary search can help in multi-objective optimization problem, especially when more metric parameters are considered. However, this is out of the scope of this dissertation.

It is a natural choice to align these utility parameters with the performance objective fitness parameters in the evolutionary search algorithm (See next section.); therefore the same set of evaluation functions can be used for both solution generation methods. Combining these utility values with similarity for the case retrieval leads to a multi-objective optimization.

4.3 Evolutionary Solution Search and Related Learning

A Genetic Algorithm (GA) is a search technique in computing to find true or approximate solutions to optimization and search problems. It is a particular class of evolutionary algorithm [111, 113] that use methods inspired by evolutionary biology such as inheritance, mutation, selection, and crossover (also called recombination). Evolution is a process that, through iterated replication-with-selection of large populations through many generations, searches out the possibilities inherent in the "physics and chemistry" of the organic medium in which it is embedded. Evolution exploits any inherent properties of the medium, and flows into natural life tides that realize or flush such properties away over time. Bearing the nature of evolutionary search, a GA emphasizes on explicit problem encoding and computational implementation of genetic operations including recombination, mutation and selection [112]. In GA the evolutionary process is viewed as the development and combining of useful schemata, and an explicitly defined set of fitness functions are defined are chosen as the as primary criteria to select better solutions. The fitness evaluation of all the individuals in the search space (solution space) forms a fitness space. The fitness definition is the bridge between solution space and fitness space. A GA takes in performance objectives and searching strategies by fitness functions and searches for an optimum directly in solution space [118]. A block diagram of a general GA is shown in Figure 4.4.

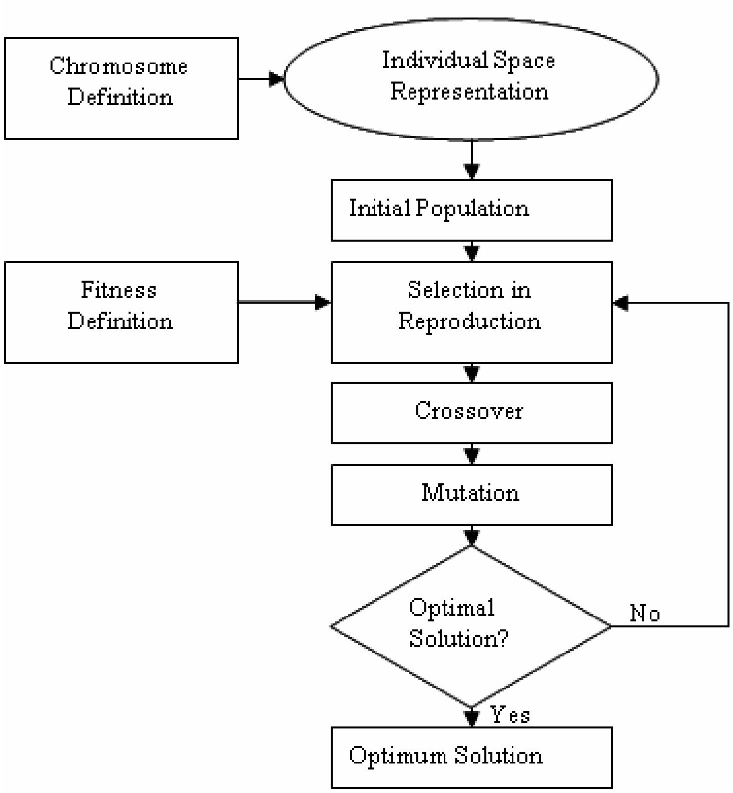


Figure 4.4: A general GA processing flow chart

The chromosomes structure in GAs allows for flexible encoding for the target problem. Taking cognitive radio as example, a parametric chromosome structure can simultaneously embody the current radio hardware configuration, radio environment condition, user service types, regulatory constraints, etc.

The population concept of the solution group in GAs enhances the optimizing force by providing multiple solutions to form a Pareto Front, which is a set of solutions with comparable overall performance evaluation, but with different trade-offs at different objective dimensions. Therefore, a solution can be extracted in a flexible manner according to secondary objectives or constraints, which makes the optimization more robust.

GAs use fitness functions as primary criteria to select better solutions in algorithm iterations. Fitness is typically defined by a set of mathematic functions containing both optimization objectives and system considerations, both of which jointly direct the evolution process. Therefore, the performance is largely determined by how well the problem is translated into the fitness function. The fitness evaluation of all the individuals in the search space forms a fitness space. The searching strategy should be adapted to suit the modality (landscape: valleys and hills which stand for local and global critical points) of the fitness space to improve the searching speed. In this dissertation, the research focus of GA for CR learning core is on applying multi-objective search strategy for radio domain, and on designing algorithm adaptation techniques to balance speed and performance.

GAs are well suited to multi-dimensional decision problems due to their parallel processing in all dimensions to find global optima as well as their ability to include constraints about the problem [119]. A Multi-Objective Genetic Algorithm (MOGA) analyzes a number of, often-competing, objectives for optimization. Reference [120] gives a comprehensive overview of the concepts and literature of multi-objective optimization problems and presents the basic formula for defining a multi-objective solution maker as shown in equation (12).

$$\begin{aligned} \min/\max\{\bar{y}\} &= f(\bar{x}) = [f_1(\bar{x}), f_2(\bar{x}),\ldots, f_n(\bar{x})] \\ \textit{subject to}:\ \bar{x} &= (x_1, x_2,\ldots, x_m)\in X \\ \bar{y} &= (y_1, y_2,\ldots, y_n)\in Y \end{aligned} \qquad (12)$$

Where there are *n* dimensions to consider in the search space and $f_n(\bar{x})$ defines the fitness function to evaluate dimension *n*. *X* and *Y* are the set of objectives and constraints respectively. The optimal solutions form the Pareto Front.

There are several important reasons why GA is suitable for the CR solution search.

(1) In wireless communication systems, the optimization objective is composite, typically including waveform format, link and network protocol, performance requirements, spectrum utilization, user service preference, and possible higher layer requirements.

Such a complicated target problem requires a global optimization method that can synthesize and handle multiple optimization objectives. The parallel nature of the genetic algorithm makes it well suited for this problem.

(2) In the radio domain, system performance requirements are coupled with operational constraints like FCC regulations, security policies and even from user orders. Although the link operation and associated performance can be modeled in a closed-form mathematical model, those regulations and policies, typically as descriptive rules, set additional logic-based objective and constraint dimensions, which are difficult to model in mathematical expressions. Unlike conventional deterministic equation-based optimization algorithms, GAs use chromosomes, which are a flexible data structure capable of including both mathematical and logical parameters, so that they can model all necessary aspects of the target problem into single hybrid optimizing process.

(3) As indicated in (1) and (2), the optimization method should be robust in finding a global optimum in the radio problem space that is usually multi-dimensional and multi-modal. Such a rigorous requirement expels most of conventional function-based algorithms such as gradient (hill-climbing) or annealing search that are could be easily trapped in local critical points, while GA is based on statistical direct search on the problem space without using any internal relation between different searching points.

As stated in Section 4.1.1, when the CR learning core encounters a novel or even totally new problem scenario, case based solution making may not be able to find a well-matched solution from past experience. Therefore, the evolutionary searching technique, specifically the genetic algorithm, is used to evolve a new solution that is more suitable for this particular problem. The GA based solution search design for CR learning core comprise three major issues, problem encoding, objective encoding and searching strategy.

Problem encoding

In optimum search, a problem is encoded by defining a set of parameters to form a general solution format. In GA, such a parametric vector is called a chromosome. For CR applications, the chromosome structure of the solution could embody the current radio platform capability, radio environment waveform format and link stack structure, propagation channel model, user service type, regulatory constraint format, etc.

The chromosome definition directly reflects the coverage of the problem. While a general solution in CR applications should occur across all layers and aspects of the radio's operation, it mostly contains PHY and MAC layers in the current GA search space implemented in the CR learning core. Future work will extend optimization up to the network, transport, and even application layers. The chromosome definition with the radio part is largely set by the available resources and processing capability of the specific radio platform, which is the CWT^2 SDR waveform framework in the CWT^2 CR node (See Chapter 6).

Performance objective encoding

In effective wireless communications, the choice of the solution (encoded as a chromosome) including PHY and MAC layers affects the radio's behavior in many dimensions such as bit error rate (BER), bandwidth, power consumption, and link latency, etc. Each of these dimensions has certain relation to the QoS, and these relations change with their relative importance depending on the application being used. These goals often compete with each other. A simple optimization scenario might call for minimizing BER and minimizing power consumption, which are competing objectives. As the radio's power consumption is decreased by turning down the transmitter power or using less optimal but more computationally efficient signal processing algorithms, the BER will increase. The only solution is to balance the power and BER to create a waveform that satisfies the conditions as well as possible. This example has only two objectives, but many more objectives exist with dependent relationships like data rate, occupied bandwidth, spectral efficiency, latency, etc. We provide a detailed analysis of the multi-objective nature of CR performance optimization in Chapter 7 of [26].

Optimization must take place in three major areas: the user/application domain, the link quality between nodes, and network interactions. When combining the optimization problem in the PHY and MAC layers, it is important both to quantify the cognitive radio interactions and measure the impact of each radio on the others. When one node is competing for spectrum resources, decisions on waveform adaptation must properly reflect and respect the needs and operations of other radios and networks in the same RF environment. Each of the optimization dimensions interact strongly with each other; changing to a more robust channel coding method may improve robustness in a bad channel, but the cost in latency may negatively affect the user's QoS. Such an interaction at waveform level optimization is illustrated in Figure 4.5. The arrows represent direct relations between these performance metrics.

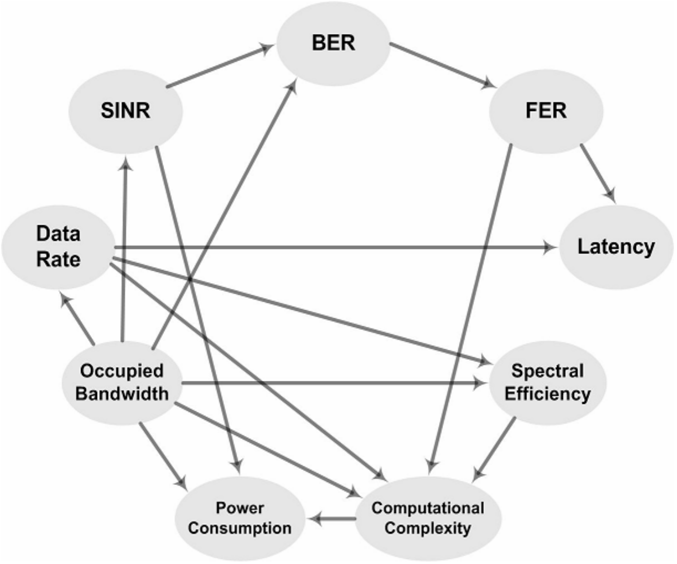


Figure 4.5: Optimization dimension interaction at waveform level

In GA search, performance objectives are defined as fitness functions [112]. The detailed description of constructing the fitness functions for PHY and MAC layer multi-objective solution search is given in our recent publication [121]. Here an example set of fitness functions is presented to illustrate the currently focused performance objective dimensions in the GA solution search. Their theoretical background can be easily found in standard communication text books [42, 51].

(1) BER for BPSK modulation signal in AWGN channel

$$P_e = Q\left(\sqrt{rT_0 B \frac{2C}{N}}\right) \tag{13}$$

(2) BER for MPSK modulation signal in AWGN channel

$$P_e = \frac{2}{\log_2 M}\left\{Q\left[\sqrt{T_0 B}\sin\left(\frac{\pi}{M}\right)\sqrt{r\frac{2C}{N}}\right]\right\} \tag{14}$$

(3) BER for MQAM modulation signal in AWGN channel

$$P_e = \left(\frac{4}{\log_2 M}\right)\left(\frac{\sqrt{M}-1}{\sqrt{M}}\right)\left\{Q\left[\sqrt{\frac{3rT_0 B}{(M-1)}\frac{C}{N}}\right]\right\} \tag{15}$$

(4) Spectrum efficiency with respect to pulse shaping roll-off factor

$$S_{eff} = \frac{k}{k(1+\alpha)} \tag{16}$$

(5) Occupied bandwidth

$$B = R_s k(1+\alpha) \tag{17}$$

(6) Data rate

$$R_b = R_s k\left(\frac{k_{FEC}}{n_{FEC}}\right)\left(\frac{T_{Tx}}{T_{Tx}+T_{Rx}}\right)\left(\frac{L}{L_{\max}}\right) \tag{18}$$

(7) Signal to interference and noise ratio (SINR)

$$SINR = \frac{T_0 C}{N + \sum_i I_i} \tag{19}$$

(8) Frame error rate

$$FER \le \sum_{m=t+1}^{n_{FEC}} \binom{n}{m} P_e^{\,m}(1-p)^{n-m} \tag{20}$$

The symbol glossary is provided by Table 4.1.

Table 4.1: Symbol glossary for example fitness functions

Symbol	**Meaning**
C	Carrier Power (W)
N	Noise Power (W)
B	Bandwidth (Hz)
T_0	Symbol Period ($1/R_s$) (second)
R_s	Symbol Rate (baud)
M	Modulation order
K	Number of bits per symbol (log2(M))
A	Pulse shaping filter roll-off factor
T_{Tx} / T_{Rx}	Amount of time to transmit / receive in a TDD system
L / L_{max}	Number of bytes / maximum bytes in a packet
R	Coding rate = k_{FEC}/n_{FEC}
k_{FEC} / n_{FEC}	Number of message / codeword bits in a block of coding
T	Number of correctible errors in a block code

Searching strategy

Figure 4.5 illustrates that one objective directly impacts the values of others that share common parameters. Therefore, the optimal solution provides the most balanced performance for the user's requirements under observed problem space. Each fitness function needs to be weighted to represent the relative importance associated with each objective, which leads the optimal search toward the Pareto Front where the solutions are called non-dominated since the optimization in one dimension negatively impacts other dimensions. The solution selection on the Pareto Front is explained in detail in [116]. Basically, there are two ways to incorporate multiple objectives into GA process:

(i) Each objective is represented by a fitness credit, and then multiple fitness credits are combined, in a way that each credit is weighted according to its specific priority, into a composite fitness which is used for the selection process in the GA.

(ii) All the required objectives are combined, also in a weighted manner, into a composite objective, which is then mapped to a fitness evaluated for the selection process.

From the mathematical point of view, these two methods are statistically equivalent. However, the GA behaves differently between these two. In case (i), the GA searches for better solutions in a multi-dimensional fitness landscape. Since each dimension of the fitness landscape is associated with some part (not complete) of each individual in the population, different recombination and mutation operators can be implemented for each part of the individual chromosomes according to the specific feature of its related fitness dimension. While in case (ii), since all the objective functions are aggregated into one big fitness calculation, thus their mutual difference with specific element fitness are cancelled before being mapped into fitness landscape. Therefore the relative diversity from the multiple fitness functions is lost, which could be taken advantage by the GA where each object fitness value is calculated individually during the search. On the other hand, case (ii) has less system complexity and its performance is easier to analyze and predict using statistical models. In (ii), reducing the dimensionality of the fitness landscape equivalently reduces the dimensionality of recombination and mutation operators.

Following method (ii), by linearly weighting the different objective dimensions according to user preference, optimization leads to a combinatorial-optimal direction to form the Pareto Front. Simulation results prove its success and usefulness in providing waveform solutions with different service requirements. The detailed performance report won an outstanding paper award in the SDRF 2004 Technical Conference [31].

While the optimization on the PHY and MAC parameters tries to build a waveform for a link to maximize the QoS, a generated solution must always obey any regulatory limitations. Spectrum and power are two areas of major concern here. Devices operating in different frequency bands are subject to different restrictions - Consider for example, those of Part 15 of the FCC specifications [122] or those governing operation in a satellite band. Certain bands would have strong restrictions governing transmitter power level (GPS), locations (TV broadcast), or time (public safety). There is a lot of discussion about opening many of these bands for other models of operation. Any operation in these bands must ensure non-interference with the primary licensed operators either through radio environment recognition (See Chapter 3.), a policy/regulatory database and language [10], or a combination of these two.

Another fundamental constrain in optimal solution search comes from the supporting radio platform. A waveform solution can not go beyond the radio's capability. As stated earlier, a radio's available processing resources generally defines the radio's operational space, and the related operational range largely sets the related value boundaries in the solution space. Such boundaries may be waveform and platform specific. For example, with the USRP radio front-end used for the CWT^2 CR system, while a solution with a wideband multi-carrier waveform might provide a high data rate and flexible spectrum occupancy, the provided hardware data interface may be not sufficient for the bandwidth proposed by the waveform solution.

A GA has the proven advantage of handling a multi-objective, constrained, global optimal solution search. Although the classic GA itself always works, its performance depends on both the searching strategy and the fitness space modality. The fitness space for CR applications is very complicated (See Section 3.1.6), and also time-varying due to the statistic nature of the radio environment. However, a GA's algorithm related performance, such as optimality, convergence speed and memory footprint, could be greatly improved by introducing an adaptation strategy during the searching process.

First, a GA can adapt its searching configuration to the target problem:

- Different searching tactics such as efficiency-driven and accuracy-driven searching rules can be applied to different radio environment and service requirements.
- A GA can also adapt its memory effects functions to make the search instant-calculation oriented or history oriented during the search iterations. For example, section pressure can be reduced by switching proportional selection to tournament selection when the searching process starts to converge so that explorative pressure needs to be increased.

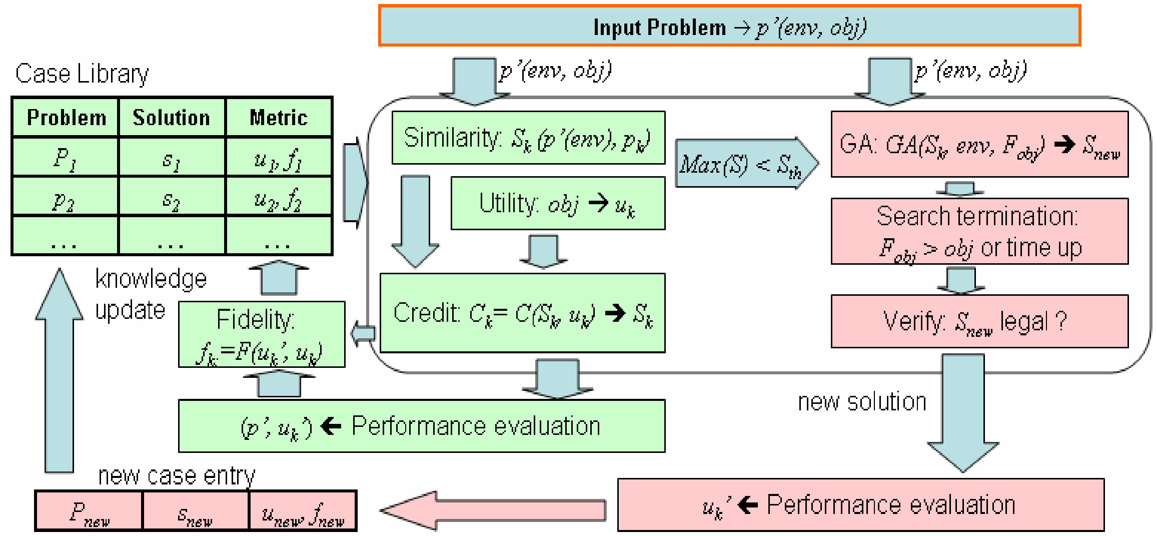
Second, a GA can adapt the searching mechanism to its own status:

- The GA can tune itself between exploitative and explorative pressure according to the iterations or converging stage.
- The GA can tune its selection pressure according to its instant population genotype (solution) or phenotype (fitness) distributions. For example, different crossover rate and mutation rate are applied to individuals with different fitness values so that superior solutions are more protected while inferior ones are more punished

For a global optimum search on a stochastic fitness space, an adaptive GA was designed that is 10 times faster than the commercial Matlab GA toolbox [123]. It is used for a dynamic spectrum allocation optimization system and made a 20dB processing gain on average. For details, see our paper [32].

4.4 CBR-GA Learning Core System Integration

Following the learning principle described in Section 4.1.1, now with both CBR and GA in hand we can integrate the CBR-GA processing chain as the learning core of the CR node. The resulting CBR-GA solution making and learning system is shown in Figure 4.6.


Case Library

Problem	Solution	Metric
p_1	s_1	u_1, f_1
p_2	s_2	u_2, f_2
…	…	…

Input Problem → $p'(env, obj)$
$p'(env, obj)$
$p'(env, obj)$
Similarity: $S_k(p'(env), p_k)$
Utility: $obj \rightarrow u_k$
Credit: $C_k = C(S_k, u_k) \rightarrow S_k$
$Max(S) < S_{th}$
GA: $GA(S_k, env, F_{obj}) \rightarrow S_{new}$
Search termination: $F_{obj} > obj$ or time up
Verify: S_{new} legal ?
knowledge update
Fidelity: $f_k = F(u_k', u_k)$
$(p', u_k') \leftarrow$ Performance evaluation
new solution
new case entry
p_{new}
s_{new}
u_{new}, f_{new}
$u_k' \leftarrow$ Performance evaluation

Figure 4.6: Information processing flow in CR machine learning

The solution making procedure starts with an incoming problem, i.e., with a statement containing the radio environment, the radio platform, and user performance objectives. This statement shares the same parametric format as the problem scenario part of the case entry stored in the CR's case library (See case definition in Section 4.1.2.). The similarity is calculated between the input problem and stored cases, and the result leads to two branches of the solution making process.

If the correlation is high between the input problem and some stored case, meaning that the problem is either the same as or very similar to one seen before, then the solution from the case with the highest similarity is deployed for practice. And the performance is recorded to update the utility metric field. Thus the case library is updated by the case with this specific input problem.

Otherwise, if the correlation is low for all the previous stored cases, which means the problem is novel or not seen before, then the GA is to run the evolutionary search for a creative solution. There are several important GA configuration issues. The initial population is fed with the solutions from those cases that have relatively stronger correlations with the problem, thus the GA is "seeded" instead of starting with a purely random initialization. The fitness functions are chosen depending on the performance objectives from the input problem. The evolutionary search is terminated by two possible criteria, *objective met* or *time up*; therefore, unnecessary computational waste is avoided. Once a new solution is made by the GA, it is verified by a legality check [121] since it is a non-standard solution. If it is legal, it is deployed to the radio practice, similarly the performance is also tracked and recorded. The input problem statement and the GA's new solution and performance metric are combined and a new case entry is created and added to the case library for future use.

It is worth emphasizing that in GA convergence control, different termination criteria can be applied for specific performance requirements. The time-up termination criterion is applied when the searching processing is limited to a fixed amount of cost range, typically in terms of time period or computational power, so the optimal solution is "as is" at the end of the search. The goal-met termination criterion is applied when the search keeps running until certain optimization criteria are reached, typically in terms of fitness values. Both methods are important for radio domain solution search. Time-up termination control is suitable for time or cost critical problem scenarios; while in goal-met termination control, the fitness reached provides a useful performance estimation reference to be compared with the real world practice, resulting in a valuable lesson to update the knowledge associated with the picked solution for future use.

With such a CBR-GA combinatorial learning core, it is clear that the rationality and creativity are both emphasized and balanced subject to the input problem. CBR is very efficient for known and similar problem scenarios. It provides standard and general solutions; thus the performance is assured. It supports learning by accumulating the knowledge with updated cases through new performance metrics upon input problems. A GA is very powerful in creating new solutions adaptive to novel problem scenarios when historical knowledge does not help. It is also useful in optimizing a solution based on current remembered standard solution when none of these solutions can meet the specific problem requirement, especially for the situation where the environment is known by a previous case, but user service demand is upgraded. The GA supports learning by simply create new case entries to enrich the case library.

Both the GA and CBR share the same solution format and performance metric set. This makes it possible for both of them to share the same knowledge base. As stated before, CBR can be viewed as pattern based knowledge system, while GA can be viewed as a domain-rule-intervened (by domain encoding and fitness definition) knowledge system. Although different in solution making mechanism, they converge on knowledge buildup based on the same reinforcement learning principle.

4.5 Radio Domain Knowledge for Reasoning and Learning

The importance of domain knowledge design for machine learning has been well explained in [26]. For our CR radio node system, the investigation of domain knowledge design is to enable and support machine reasoning and learning for radio applications. It can be divided into two steps: the first is to pick efficient methodologies of observing and modeling useful domain information and abstract such information with a general representation for computational reasoning and learning, such as CBR-GA learning core in our CR node system; the second is to design a powerful language to describe such domain knowledge for information transferring between intelligent functional units.

The modeling and representation of domain knowledge is called a profile (See the detailed ontological definition in Chapter 12 of [26].). The format of a domain knowledge profile depends on the modeling method and characteristics for the specific domain [124]. Understanding that the CR system sits at the center of the radio, user and policy domains, we define the following domain knowledge profiles at the currently developmental stage of our CR node.

(i) Radio environment profile: contains the information recognized from the radio environment, including spectrum energy occupancy, waveform format at channels of interest, and available link opportunities.

(ii) Radio platform source profile: contains the list of available radio functional resources, and their related processing capability in terms of dynamic range and working modes.

(iii) Waveform profile: contains the communications level representation of a waveform specification, such as desired carrier frequency, modulation scheme, pulse shape, coding scheme, frame format, MAC protocol, etc. The waveform profile gives a complete definition of a waveform that can be implemented on different radio platforms.

(iv) Radio platform configuration profile: contains the configuration parameter set of the radio platform to carry required waveform and link operations. Such a configuration is platform specific and the settings inside depends on the required radio functionalities.

(v) Performance profile: contains a set of metric parameters as the evaluation of radio performance, such as BER, PER, SINR, data throughput, power consumption, etc. These metrics are platform independent and thus can be applied to different radio hardware system.

(vi) User service profile: contains a set of metric parameters as the performance objectives in guiding radio's operation. These metrics are derived from the user's service demands, such as desired service type, data rate, service quality, etc.

(vii) Policy profile: contains local regulatory information that the radio needs to conform in the field, such as frequency plan, power limitation, spectrum shape, interference to adjacent channels, etc. This profile is used to provide legality verification of the solution made by the CR learning core.

Representing domain knowledge in the form of domain profiles ensures that a general machine reasoning and learning structure can be designed and applied across multiple domains. It enables the independence of information processing methodology from the specific information content. In the CWT^2 CR learning core, the CBR-GA learning structure works on the radio, user, and policy domains via the understanding and generation of the domain profiles listed above.

As a profile is defined to stand for the domain knowledge, it needs to be described in a standard data format to transfer such information between different processing units in a cognitive system. Such a profile describing format is termed a language in our cognitive radio system design. An artificial cognition design is based on the information processing structure of a computational system [125], in which cognitive components are divided into several independent modules like sensors, central information processor dealing with reasoning and solution making, and practice adaptation system. The key assumptions of this computational structure are that the information processing is a symbolic system and the modules function independently. Language can be viewed as a separate interface component to such processing.

Since the language is the vehicle of the domain profile, the key challenge of profile design, which is to synthesize the massive, discrete information collected directly from domain observation, is translated to the language design to come up with a script with a data structure that not only remains generally applicable for various information processing interfaces but which also bears a resemblance in parametric format to support machine logic and calculation. It also should be legibly descriptive for human understanding.

To achieve a good balance between speed and flexibility, we choose Extensible Markup Language (XML) to describe the domain profile. XML was developed in 1996, simple in structure and concise in content. It has been used by a wide variety of applications, and is human legible. A general introduction of XML standard is given by [126]. XML provides a generic solution to information connectivity between programs that were previously difficult to interface. It follows the Unified Modeling Language (UML) [127] modeling standard and provides object-oriented, self-explanatory scripting in which we can flexibly grow and define data structures as well as data content. This scripting is standard and can be realized regardless of programming language, i.e. with C/C++, Python, Java, etc. Therefore a machine reads XML data by a simple data parsing process.

Inside the CWT^2 CR node system, XML serves as a method of representing data between functional modules. The radio platform passes information to and from the cognitive engine in XML, containing the profiles such as radio resource, radio environment, and radio performance. The cognitive engine tells the radio in XML the profiles of the waveform to implement, the radio operational setting, etc. The CR knowledge base is applied and evolved by the information in XML as well. Currently the cognitive engine is shifting from a node based implementation to network based distributed implementation where all the knowledge profiles' information is shared and transferred via XML. A complete CR node system with all knowledge profiles described by XML is detailed in Chapter 7.

4.6 Summary

Cognitive service requirements are exceeding today's reconfiguration and adaptation methods, and thus they call for the machine learning capability to enable the radio to understand its environment, including user needs, and optimize its performance. Following the reinforcement learning principle, the Case Based Reasoning (CBR) technique is used to make solutions. CBR relies on a knowledge based reasoning approach in which the solution making relies on the past experience. Although effective with familiar problem scenarios, CBR performs poor in a new situation. When the association between the encountered problem and previous experience is difficult to generate, a more "creative" solution making mechanism is incorporated using a GA. With powerful problem encoding, the GA can effectively provide multi-objective solution search on a complicated, unfamiliar problem space.

In the CWT^2 CR node system, the machine learning core is the combination of CBR and GA to balance between rationality and creativity. The solution making strategy is adapted according to the familiarity to the encountered problem scenario. Once the solution is made, it is put into practice, and its performance in the real world is recorded as the solution’s metric for future reference. Thus with its knowledge updated the CR learns from its own experience. In the radio knowledge design for the learning core, a case library is implemented with a common solution format for both CBR and GA.

Chapter 5: General Radio Interface

5.1 Platform Independent Radio Knowledge Interface Design

The importance and concept of a platform independent radio interface for the cognitive engine is explained in Section 2.1.2, and the interface structure is also illustrated in Figure 2.2. Generally speaking, the radio interface is responsible for delivering platform knowledge to the CE to form its basic operational space and for carrying configuration and control from the CE to instruct the radio's specific operation adaptation.

If a CE wants to control the radio, it must first know the radio's capabilities. It would be worthless to create a modulation setting not supported by the radio. When optimizing a waveform, the radio capabilities are part of the constraints of the search space. We must therefore have representation of the radio (i.e. the radio platform profile, Section 4.5). Such a representation abstracts the radio's implementation details and looks solely at the communication level [128]. Thus it can be used to represent various radio platforms regardless of architecture and hardware realization. In this way, without knowing how a specific modulation or channel coding is implemented inside the radio, we only need to know what modulations or channel coding algorithms are available.

On problem might occur when certain elements of a waveform are mutually exclusive of others. In an ideal SDR platform, all modulations and all channel coding algorithms can work with each other, but perhaps a specific SDR only allows certain block sizes with a modulation of a certain order. When learning the radio's capabilities, if this situation exists, it must be properly represented in the platform profile.

To actually get the information about the radio, we have two methods. First, the cognitive engine can actively query the radio platform and ask for its resource and capability report.

This is some form of a service discovery protocol such as a Peripheral Component Interconnect (PCI) bus or Universal Plug and Play (uPnP). The drawback to this method is that it requires a universal standard for all the radio platforms.

Another method is much simpler, a manually-coded representation by the radio designer or manufacturer. Although this is not an elegant solution, it avoids an additional protocol layer between the radio platform and cognitive algorithms, which is very helpful in improving system efficiency and real-time response, especially for CR applications. This platform profile provided by the radio maker helps the cognitive engine form a more confident operational space than ad-hoc querying and machine understanding. The content (such as parameter values) in the radio profile can be coded in a totally device specific manner as far as the profiles of different radios share the same data structure. Therefore, there is no need (and practically it is very difficult) to ask various radio platforms to obey the exact same performance evaluation standard. Normally, at one CR node during one operational session, it is unlikely for the cognitive engine to switch from one platform to another. Therefore, this platform knowledge pre-loading approach is suitable. Radio capabilities do not change fast, and so the listing of these capabilities will be done once by the radio developer, with periodic updates that will correspond to device or software upgrades.

In the radio knowledge profile, design follows the object oriented approach in modeling the radio platform. Each processing resource element is viewed as an object. Its related characteristics, such as working modes, processing range, data I/O format, and computational cost are coded as the attributes of this object. The trick here is to use a representation language, XML to describe this model and for CE to parse. A simple example is given here.

XML formats data in a tree, where we start with a root node, usually a high-level construction like <radio> with branches that represent processing objects of the system. To represent a radio in XML, the radio node first branches off into receiver and transmitter branches; from here, the tree branches into components like RF front-end and modem with different working modes and operational ranges that the radio supports. The leaf nodes contain data, or values, associated with each component, such as a carrier frequency. To make the language even more powerful in its representation, each node may contain a set of attributes. In the example of specifying a carrier frequency, an attribute might list the units used. The XML document could specify the carrier frequency as from 2300e6 to 2900e6 and a tuning resolution of 1e3with an attribute of units set to "MHz". The parser retrieves both the value and the attribute of the node, which is then simply translated by the cognitive engine into usable information about the carrier frequency. In the tree structure of XML, a parser can be completely agnostic of the data format of the XML document, and the cognitive radio can easily "walk" through the tree to find components, values, and attributes of the component. A partial XML file for this example is shown below:

```
<?xml version="1.0"?>
<!DOCTYPE WAVEFORM SYSTEM "gnuradio_hw.dtd">
<radio waveform type="analog/digital">
       <Tx>
           <PHY>
                          <rf>
                              <tx_freq units="Hz" mult="1">
                                   <min>2300e6</min>
                                   <max>2900e6</max>
                                   <step>1e3</step>
                              </tx_freq>
                              <tx_power units="mW" mult="0.001">
                                   <min>1</min>
                                   <max>20</max>
                                   <step>0.1</step>
                              </tx_power>
                          </rf>
                          <rf>
                                   …
                          </rf>
                          <mod>
                                  <tx_mod type="PSK">
                                   <tx_mod_bits>
                                             <min>1</min>
                                             <max>3</max>
                                             <step>1</step>
                                   </tx_mod_bits>
                               <tx_mod_differential>
```

```
                                           <min>1</min>
                                              <max>1</max>
                                              <step>0</step>
                                    </tx_mod_differential>
                                    </tx_mod>
                                    <tx_rolloff units="na">
                                              <min>0</min>
                                              <max>1</max>
                                              <step>0.01</step>
                                    </tx_rolloff>
                                    <tx_bt units="na">
                                              <min>0</min>
                                              <max>1</max>
                                              <step>0.01</step>
                                    </tx_bt>
                                    <tx_gray_code>
                                              <min>0</min>
                                              <max>1</max>
                                              <step>1</step>
                                    </tx_gray_code>
                                    <tx_symbol_rate units="Hz" mult="1">
                                              <min>15000</min>
                                              <max>400000</max>
                                              <step>10000</step>
                                    </tx_symbol_rate>
                          </mod>
                 </PHY>
                 <LINK>
                          …
                 </LINK>
       </Tx>
       <Rx>
                 …
       </Rx>
</waveform>
```

With this radio platform modeled in XML, the CE can apply simple parsing and data format checking methods to extract operational parameters to construct the solution space used by the CR solution making and learning core. A complete radio platform profile in XML implemented in the CWT^2 CR node system is provided in Appendix B.

5.2 Platform Independent Radio Configuration and Control Interface Design

In the cognition cycle shown in Figure 2.6, the processing flow starts and ends with the radio platform. The cognitive engine must first know the radio platform's capabilities, to setup the basis of solution space, then make waveform (currently PHY and MAC) solutions by reasoning knowledge profiles from different domains, including the radio platform. The solution is then passed back to the radio platform via the configuration and control interface between CE and radio.

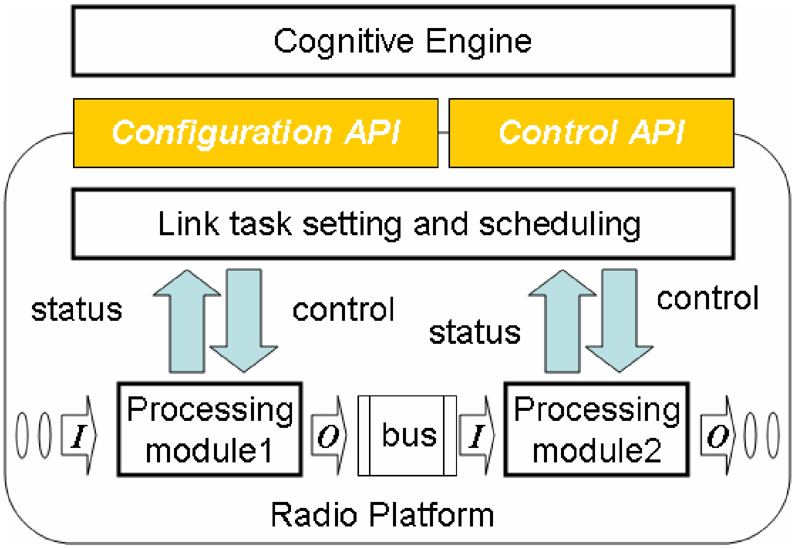


Figure 5.1: Radio platform configuration and control interface

Such a configuration and control interface from CE to radio is illustrated in Figure 5.1. The radio takes in both the configuration and control instructions from the CE with its provided API set and control logic respectively. These APIs and control logics are already registered as radio resources in the platform knowledge profile loaded to the CE. Therefore, one configuration and control instruction set contained in a solution from CE is simply one specific setting of these radio resources.

More specifically, in the current development stage of the CWT^2 CR node system, the cognitive capability mainly covers PHY and MAC layers. The solution made by the CE focuses on the waveform that is to be deployed in the radio platform. Therefore, a solution consists of two parts, waveform profile and radio configuration profile. They jointly determine the configuration and control of a specific link session to be launched. One example waveform profile XML and its associated radio configuration profile XML are displayed below.

```
<?xml version="1.0" encoding="utf-8"?>
<waveform type="digital">
   <MAC>
      …
   </MAC>
   <Tx>
      <PHY>
         <rf>
            <tx_freq range="100e6,500e6" units="Hz">462662500</tx_freq>
            <tx_power range="0,100" units="mW">0.1</tx_power>
         </rf>
         <mod type="psk">
            <tx_mod_bits>1</tx_mod_bits>
            <tx_mod_differential>1</tx_mod_differential>
            <tx_roll_off range="0,1" units="na">0.35</tx_roll_off>
            <tx_gray_code>1</tx_gray_code>
            <tx_symbol_rate range="10e3,500e3"
units="symbols/s">350000</tx_symbol_rate>
         </mod>
      </PHY>
      <LINK>
         <frame>
            <tx_pkt_size range="1,1500" units="bytes">512</tx_pkt_size>
            <tx_access_code>None</tx_access_code>
         </frame>
      </LINK>
   </Tx>
   <Rx>
      …
   </Rx>
</waveform>

<?xml version="1.0" encoding="utf-8"?>
<radio_platform type="GNU Radio">
   <Debug>
```

```
      …
   </Debug>
   <Radio>
      <tx_usrp_subdev_spec>(0,0)</tx_usrp_subdev_spec>
      <rx_usrp_subdev_spec>(0,0)</rx_usrp_subdev_spec>
      <tx_usrp_pga_gain_scaling>1.0</tx_usrp_pga_gain_scaling>
      <rx_usrp_pga_gain_scaling>0.5</rx_usrp_pga_gain_scaling>
      <tx_samples_per_symbol>4</tx_samples_per_symbol>
      <rx_samples_per_symbol>4</rx_samples_per_symbol>
      <control>
          <interface>all</interface>
          <port>5556</port>
          <socket_buffer_size>1024</socket_buffer_size>
      </control>
   </Radio>
   <NET>
      <interface>gr0</interface>
      <IP>192.168.200.2</IP>
      <socket_buffer_size>10240</socket_buffer_size>
   </NET>
   <MAC>
      …
   </MAC>
   <SRC>
      <file>music-james-blunt-you-are-beautiful.dat</file>
   </SRC>
   <SINK>
         <NET> … </NET>
   </SINK>
</radio_platform>
```

Note that the waveform profile is platform independent so it can be generally applicable for different radio platforms with standard XML parsing capability. An example waveform interface processing flow at the radio side is illustrated in Figure 5.2. In this flow a translation occurs from general waveform representation to the radio-specific implementation to control the radio’s operation. Specifically, this parsing structure is implemented with GNU Radio SDR system. A detailed report is given in [129].

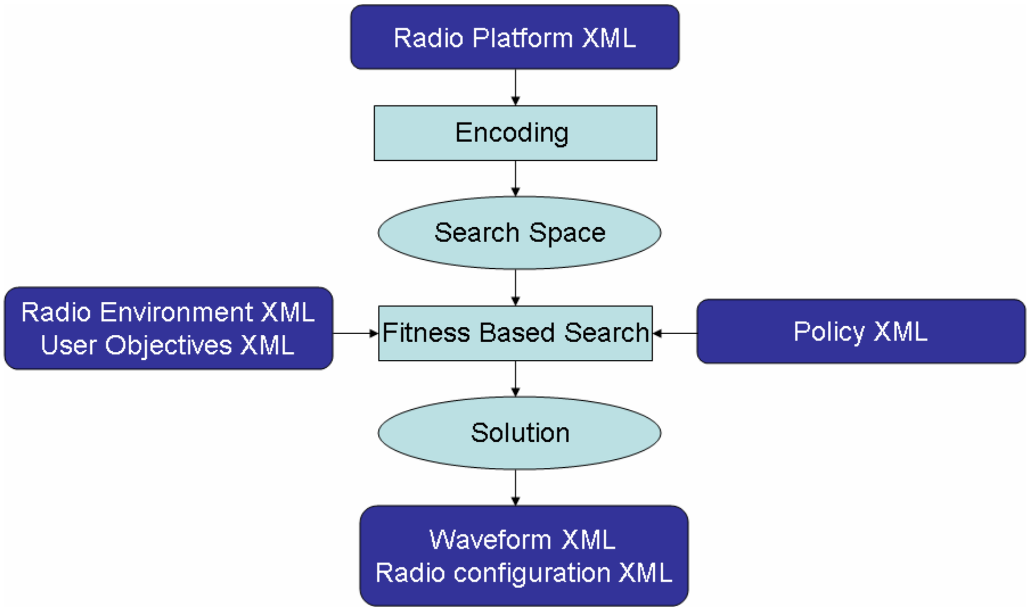


Figure 5.2: Example radio waveform interface processing flow

A radio can be viewed as a device with sets of knobs and meters. The CE reads the meters to obtain radio environment and radio platform information and turns the knobs to control the operation. We explicitly define interface as the set of software functions, documents, and additional application tools required for the CE to configure and control the radio platform. The API is the set of functions used by the radio platform to invoke the operations required from the interface. The API can be considered as the sets of knobs and meters provided by the radio platform for CE to configure radio's waveform setting and control the radio's processing behavior. Following the platform independent design principle, the "knobs" and "meters" view of radio API is illustrated in Figure 5.3.

Currently a GNU Radio and USRP based software defined radio is used as the radio platform for the CWT^2 CR node. The currently supported knobs and meters are the ones in bold listed in Table 5.1.

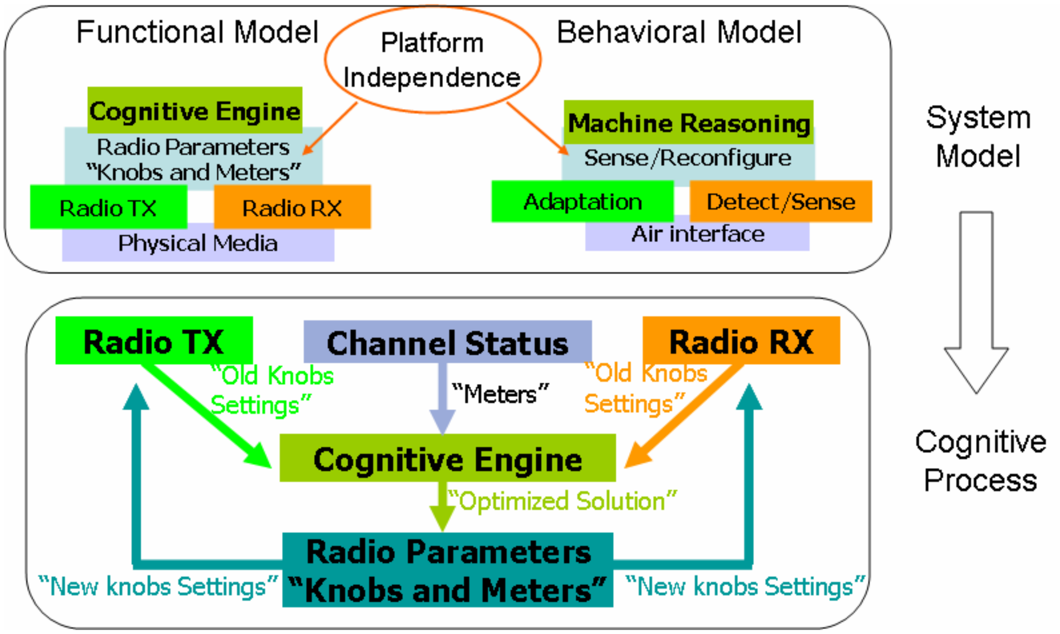


Figure 5.3: Platform independent configuration API – "knobs" and "meters"

Table 5.1: Symbol glossary for example fitness functions

Layer	Meters	Knobs
NET	Packet delay	**Packet size**
	Packet jitter	Packet rate
MAC	**CRC check**	**Source coding**
	ARQ	**Channel coding rate and type**
	Frame error rate	**Frame size and type**
	Data rate	**Interleaving details**
		Channel/slot/code allocation
		Duplexing
		Multiple access
		Encryption
PHY	**BER**	**Transmitter power**
	SNR	Spreading type and code
	SINR	**Modulation type**

	RSSI	**Modulation index**
	Path loss	**Bandwidth**
	Fading statistics	**Pulse shaping**
	Doppler spread	**Symbol rate**
	Delay spread	**Carrier frequency**
	Multipath profile (OFDM)	Dynamic range
	AOA	**Equalization (OFDM)**
	Noise power	Antenna beam form
	Interference power	
	Peak-to-average power ratio	
	Error vector magnitude	
	Spectral Efficiency	

While the configuration API can be viewed as a set of knobs and meters, the radio control API is for the radio platform's operational behavior, i.e., duplexing control, data frame timing, MAC timing control, state switching control with specific mode modes, and working mode adaptation control. To support a platform independent control API, the radio's behavior can be viewed as a set of hierarchical finite-state-machines. The control API consists of both the choice of the session algorithm and its related settings. These session algorithms are determined by both PHY and MAC settings given by the waveform profile XML from CE. Generally, a radio is running multi-threading processing for a waveform / link session, different waveform parameters at different communication layers adjust the radio's corresponding level of working threads, such as digital signal processing threads at PHY layer versus packet header lookup threads at network layer; and different algorithm settings trigger different level of events along with these working threads, such as RF emission switching versus packet switching. The CE only needs to tell the radio what PHY/MAC specifications the radio needs to deploy by informing both the configuration and the control APIs, and the radio will initialize specific session threads to create a link session (such as transmitting a group of packets, or broadcast several seconds of music, etc). The control APIs are set by both the waveform profile and the radio configuration profiles.

It is important to point out that in this platform independent interface, the configuration and control APIs are independent threads in radio operation. This greatly enhances the flexibility for CE to instruct the radio to carry one thread without affecting the other. In other words, waveform settings can be reconfigured in real time while the service session keeps running, therefore, and on-line reconfigurability is supported between PHY and MAC by the current general radio interface. The multi-threading design of radio session control will be explained in the CWT^2 SDR platform development in Chapter 6.

5.3 Platform Independent Radio Interface Integration

As both configuration API and control API are defined for the radio platform, the CE can form a general interface to interact with different radios. A system-level block diagram of the general radio interface is shown in Figure 5.4. Detailed explanation is given in [116].

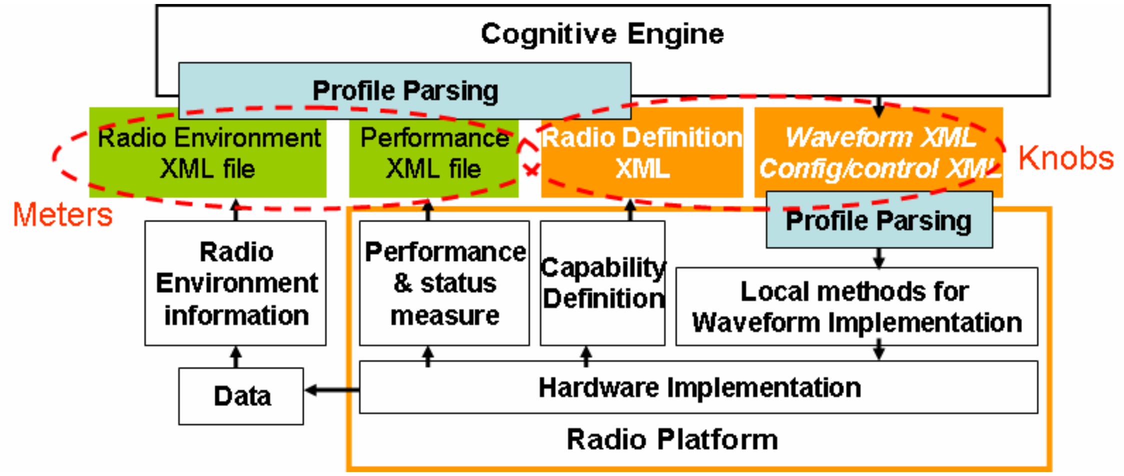


Figure 5.4: General CE-radio interface system block diagram

The CE controls the radio to conduct sensing and analyze collected data to achieve environment recognition. The resulting radio environment profile is described in the radio environment XML and passed to CE. The radio platform knowledge profile is loaded to CE by the radio definition XML, while its status and performance are reported to CE via the performance XML. The CE makes the solution on all this knowledge according to user's performance requirement, and it formulates the solution in both the waveform profile and the radio configuration profile that are passed to the radio platform via two corresponding XML scripts. The radio environment and performance XML files mainly contain "meters" that CE needs to read. The radio definition XML basically tells the CE what knobs are available from the radio, and their effective turning range. The waveform XML and radio configuration XML files jointly set particular values of the knobs used for the required waveform generation.

5.4 Summary

The radio interface is responsible for delivering platform knowledge to the CE to form its basis of operational space, and carrying configuration and control information from the CE to instruct the radio's specific operation adaptation. Radio platform knowledge is representation as a communication-level resource profile which shields the implementation details. Thus it can be used to represent different radio hardware. A manually-coded platform profile by the radio designer or manufacturer is preferred to minimize processing complexity in CE. It is also easy to maintain and update. Following UML modeling, processing resource are objects with operational attributes such as tunable parameters and tuning range, and described in XML meta-data format.

The radio takes in both the configuration and control instructions from the CE with its configuration and control API sets. One configuration and control instruction set from a CE solution is simply one specific setting instance of these radio resources.

A CE solution consists of two parts, waveform profile and radio configuration profile. They jointly determine the configuration and control of a specific link session to be launched. The configuration API can be viewed as knobs and meters, while the radio radio control APIs directs the radio's behavior, which consists of both the choice of the session/thread algorithm and the triggering events. Configuration APIs are set by the waveform profile XML from the CE, which mainly covers the PHY and MAC layers. The control APIs are set by both waveform profile and radio configuration profile XML files. Configuration and control between CE and radio are independent threads so that waveform adaptation and the application service are managed asynchronously without affecting each other.

Chapter 6: The CWT^2 SDR Platform for a Cognitive Radio Node

6.1 SDR for Cognitive Radio System: General Design Issues

6.1.1 Why SDR for a CR system

Classical radios were hardware based and fixed mode; that is, they were built from components that each had a specific purpose to realize the design goals of the radio. These worked very well, but with limited application. As digital signal processing became fast enough, portions of the radio were migrated into software. Progress pushed more and more of the communications signal processing into chips and microcontrollers, and we now have a very fruitful field of work in Software Defined Radios (SDRs).

An SDR can be described as a transceiver in which much of the PHY layer is realized in software, allowing the device to be reconfigured to meet changing needs. The key to this ability to reconfigure is the support for various modulation schemes, frequency adaptation and portable waveforms. The International Telecommunications Union – Radiocommunications (ITU-R) defines an SDR as, “A radio in which the RF operating parameters of frequency range, modulation type, and/or output power can be set or altered by software, or the technique by which this is achieved” [130]. At the protocol layers we have the ability to adapt the media access control and routing capabilities, although such higher layer functionality is generally not considered part of an SDR. The foundation of SDR is rooted in a collection of hardware and software sharing the same objective of moving as much of the radio’s processing from the analog to the digital domain, from hardware circuitry to programmable codes, in order to improve the flexibility, interoperability, and efficiency.

SDR technology [131] represents a revolution in radio design philosophy. The software based design approach reduces the content of analog components and provides the highest level of function flexibility by programmable algorithms. It has been widely embraced as the way to develop frequency, waveform, and protocol agile radio platforms. It is currently best preferred as the supporting radio platform to provide flexible waveform support and link control for adaptive and cognitive radio and networking functionalities [132].

6.1.2 Challenges in SDR Design for CR Applications

For supporting radio platform, CR system requires both hardware device engineering and signal processing algorithms design to face stricter requirements than in conventional radio design. It is important to understand these key design challenges to support more than just standard radio functionalities, such as:

(i) Sweep wideband spectrum and detect low energy signals with different channel bandwidths and support adaptive channel filtering;

(ii) Recognize waveform formats at the channel of interest, decode different protocols and establish the link if required;

(iii) Acquire unknown propagation channels and equalize adaptively if necessary;

(iv) Switch flexibly between different waveforms, different working modes, and different bands for cognitive interoperability and QoS;

(v) Self-monitor, self-calibrate, and self-manage its own resources (both hardware and software) to optimize performance.

CR requires a high sensitivity from the radio front-end to achieve reliable spectrum sensing, especially to detect a weak wide-band signals in the presence of strong narrow-band signals/interference. This impacts the dynamic range requirement of the analog parts. However, noise suppression and additional processing gain can be achieved by correlation and filtering in the digital domain, given an analog signal is digitized at fractional-symbol level. In the signal detector design in the CWT^2 SDR, hierarchical windowed FFTs and overlapped sliding averaging Welch periodogram are jointly used to maximize detection sensitivity and stability (See Chapter 3.).

CR also needs the transceiver to have wide-band linearity to support various waveform modulations with little distortion. This is especially important for waveform recognition and other waveform-agile operations and different from standard-specific radios. It typically requires a high-speed high-resolution ADC, which is unfortunately power hungry and costly [35]. Thanks to digital signal processing, after a good band-pass filter, a signal can be sub-sampled into baseband, which frees the ADC from the Nyquist sampling rate at the radio frequency. Also, digital compensation can be applied to calibrate and correct the processing imperfections of low-cost devices such as mixer nonlinearity and tuning frequency drifting. Waveform features, such as amplitude and phase characteristics, are sensitive to frequency offset and result in crosstalk between orthogonal components. Frequency drifting might make the CR unable to recognize the waveform and channel correctly. It is preferred to move the digital interface close to analog down-conversion and apply digital frequency tracking. With a complex digital sample format, the image problem is eliminated in frequency conversion, as used in the CWT^2 SDR system design. For waveform recognition, an SNR-adaptive block processing algorithm is designed to combat the high noise figure from the low-cost USRP radio front-end (See Chapter 3).

Although SDR can implement much of the signal processing tasks in software, it will have to use analog parts at the antenna. At the radio front-end, linearity, bandwidth, and noise figure are mostly set by power and device cost. Front-end components have the least reconfigurability since they are mostly analog lumped components. Although performance degradation can be compensated by digital algorithms, processing consumes power, causes delay, and requires additional silicon area. The balance between hardware and computation cost should be cautiously set for specific application requirements. A little over-budget design for hardware, especially the front-end, for near-future upgrade compatibility is preferred.

A CR needs the radio to be highly flexible in reconfiguring its functions because a higher degree of freedom in adaptation gives the CR more space to optimize its performance. It is more difficult to increase the flexibility in analog parts of the radio due to their hardware nature; the major design trade-off lies between linearity and power. DSP algorithms are always reconfigurable due to their software nature – the major design trade-off is speed vs. power. For the analog part, the major design trade-off is between linearity and power [36]. Another important capability that the CR needs is the on-line reconfigurability so the adaptation can be done whenever needed and fast enough to keep up with the changing environment.

To design the SDR to support the strict demands from a CR system, it is important to identify the sweet spot of the trade space including key design dimensions including power, memory, bandwidth, and form factor, to meet the specific application needs [133].

For base stations or access points, the relatively rich budget in both hardware and processing can aim for a system level reconfigurability to fully support cognitive networking capability. The radio front-end is preferred to be all-band, wide-bandwidth, linear, and transparent as the bridge between different air interfaces. The baseband processor is preferred to be able to handle various waveform standards and network protocols. It also should have enough redundant processing power to monitor the performance, carry cognitive information processing and data mining/sharing across the network.

For mobile handheld radios where both power and computation are limited, the radio front-end is mostly designed to support a limited number of bands and waveform groups. The baseband reconfiguration can be as flexible, but such generality almost always induces relatively significant extra power and memory. To support cognitive networking, the handheld is preferred to have multi-band, multi-mode waveform reconfigurability, and focus on low-level, node-wise intelligence in optimizing its operational cost for required service, such as transmission power saving, data rate adaptation, flexible handover, etc. It is preferred to offload the network level cognitive decision making tasks to base stations. Therefore, an unbalanced cognitive system can be spread over the network to achieve maximal flexibility across heterogeneous SDR platforms and service demands (See Chapter 11 in [26].). In the CWT^2 CR node design, the same hardware platform is used for cognitive applications as both a gateway and a terminal. Therefore, though using the same analog front-end, the software algorithms are taking over on the signal processing chain as soon as complex digital low IF samples are available, and are configured differently for the best trade-off under different working modes (See Chapter 3 and the following sections of this chapter.).

As the bridge between analog and digital processing, the ADC is the key to setting the sweet spot of the whole signal processing system because (1) its resolution and bandwidth reflect the analog dynamic range of the front-end before it, and (2) its speed and linearity set the base rate of the following digital signal processing chain. To support flexible and wideband waveform conversion, the ADC needs to improve both its bandwidth and speed, which leads to significant increase in power and price [134]. Picking the right ADC is most important in defining the receiver architecture, and its budget should always couple with algorithms designed to either compensate for imperfections or relax hard limitations. The digital to analog converter (DAC) at the transmitter side basically has the same tradeoff dimensions but with a different balance point: bandwidth is more important than dynamic range for waveform flexibility, and spurious response requirements are strict to minimize spectrum leakage. In developing the CWT^2 SDR system, a deep ADC performance analysis and a comprehensive survey was conducted [35].

For digital baseband processing, whether implemented on Digital Signal Processors (DSPs), Field Programmable Gate Arrays (FPGAs), or General Purpose Processors (GPPs) or on combinations of these, real time waveform processing is always the primarily limiting factor for software defined implementation, especially for extending waveforms from narrowband analog to wideband digital modulations. Based on current semiconductors and algorithms, to implement wideband, high data rate waveforms, FPGAs are preferred over DSPs and GPPs due to speed advantage, although at a much higher power consumption. For FPGA, the functional algorithm, VHDL description, and synthesis tool efficiency will jointly determine the chip clock frequency. This is different from DSP or GPP, where the clock is fixed regardless of the code running on the chip. However, an FPGA is not as flexible in reconfiguration. Especially for an on-line mode switch, an FPGA needs to flush the gate logic and reload another one, while in DSP and GPP only a pointer jump in the memory is needed. This does not affect the work flow. In terms of speed, DSPs are merely enough for narrowband waveforms, but with lower power. And more importantly, DSPs run high-level compiled language, which is much easier to program for fixes and upgrades, speeding the development cycle. GPPs are normally used for protocol processing in commercial products, but they are becoming more popular for waveform processing due to speed improvement. They are especially useful for prototyping SDR testbeds for research or development. Recently we are seeing more baseband dependency on DSPs and GPPs while relying on FPGAs for only heavy duty sample-level work shared by many waveforms in applications like filtering, decimation and interpolation, and digital frequency conversion. Thus, “soft core” intellectual property is emerging in the communication industry to cope with reconfigurability and integration challenges [135]. Soft cores are high-level languages that are transferable between foundry processes. Soft cores such as DSP, GPP and dedicated functions are facilitating communications systems expertise to develop highly integrated software radio system with less engineering cost. Largely limited by clock rate, GPP processing capability is constantly increasing to meet real-time response and precise timing in the process control. And the availability of multi-core processing has leapfrogged the GPP based processing potential. Co-processing can significantly help to solve the data bus bottleneck and improve clock efficiency in DSP and GPP based processing structure [136]. On promising recent development of multi-core GPP is the Cell Broadband Engine processor from IBM [137]. Our CWT^2 CR node system has been built on x86 GPP platform with general Linux Operation System (OS) environment. A hardware configuration with an Intel Dual Core processor plus a a one gigabyte memory is enough to support a fully functional PSCR node (See Chapter 7). Recently we have also demonstrated the use of the GNU Radio and CWT^2 SDR platform on the Cell processor in a Sony PlayStation 3, and we are currently porting our developing CE software system to the Cell processor platform.

6.1.3 Some Existing SDR Systems

In the history of SDR development, several architectures have been developed by industry and academia. This dissertation provides a discussion of the most recent platforms that are mostly widely available and of direct interest to CWT^2 cognitive radio research. It does not include the well known SDR platforms from Vanu, Inc., the Digital Modular Radio (DMR) from General Dynamics or DARPA's SPEAKEasy program. A comprehensive introduction to these SDR systems is given in [131].

GNU Radio

One of the largest scale software defined radio projects today is the open source GNU Radio [34], a project of the Free Software Foundation (FSF). This project's goal is to provide a software radio platform that will satisfy many needs of government, industry, academia, and amateur radio hobbyists. With open architecture, open source code, and knowledgeable developers around the world, the available functionality grows quickly. Because it is based in GPP architecture, it is flexible, extensible, and portable. Currently, the GNU Radio software capabilities support the development of many waveforms including AM and FM analog waveforms and narrowband digital waveforms of GMSK, BPSK, and QPSK. Multi-carrier OFDM waveform with several MHz bandwidth is currently under development.

GNU Radio is written in both C++ and Python, and programs can be compiled and run on most general purpose processors (GPPs) and operating systems (including Linux, Mac OSX, and Windows XP). GNU Radio is typically used with a radio front-end called USRP [138], which is low-cost and open source. The software platform developed in the CWT^2 CR node system is based on GNU Radio package and uses USRP as its radio front-end. The design and implementation experience are the focus of this chapter, and the discussion of the GNU/USRP platform will continue following a brief look at some other representative systems.

Trinity College, Dublin's IRIS SDR

The Centre for Telecommunications Value-Chain Research (CTVR) of Trinity College, Dublin, Ireland is developing another software defined radio, called Implementing Radio in Software (IRIS), based on operation with general purpose processors [24]. They, too, use the USRP as the air interface to transmit data to and from the host computer, but they are moving towards another platform designed for their purposes.

The IRIS supports many of the same signal processing capabilities as the GNU Radio, including many different modulations and a working OFDM waveform. One of the most attractive features of this SDR platform is that it follows the component based software development approach. With standard and hierarchical modular interface it is completely reconfigurable at runtime with waveforms described in XML [25].

Collaborative work between the CWT and CTVR has already shown interoperability and coexistence of different SDRs. Communications links can be established between the two platforms using narrowband waveforms. The coexistence between narrow band channel and adaptively notched OFDM wideband channel also prove a mutual zero BER interference avoidance. The detailed report is given by [23].

Kansas University Agile Radio

Kansas University is building their own SDR platform that incorporates both the host baseband processor and RF front-end. The Kansas University Agile Radio (KUAR) is a compact combination of both digital and RF processing components splits the signal processing among its internal elements, including an FPGA, a general purpose processor, and dedicated hardware to transmit and receive in the 5-6 GHz band. Where much of the signal processing task is done in the FPGA, the controlling logic in the GPP allows for radio adaptation and reconfiguration. The specifics of this SDR platform can be found in [139].

SCA-based SDR Platforms

Besides research or academic SDR platforms listed above, the Software Communications Architecture (SCA) is also an important platform and close to cognitive radio technology. The SCA grew out of the military's desire for interoperable software radios and the Joint Tactical Radio System (JTRS) program [41]. The SCA is an open architecture to specify how software radios should be developed that will both meet the military's needs and provide interoperability between any radios designed with SCA standard. For the JTRS design, the hardware core interface is the particular piece of SCA that is used to pass information from the developer about the desired waveforms to the low-level drivers of the hardware.

Although the JTRS program has been declining in scope, the SCA has developed a large corporate presence, and years of SCA development have begun to produce frameworks and SDR applications. The SCA includes a core framework, an operating system, and the Common Object Request Broker Architecture (CORBA). The latter two concepts have general implementations in many formats, and so the core framework of the SCA is where much of the communications work has focused. Among the core frameworks being developed are the Open Source SCA Implementation: Embedded (OSSIE) from the Mobile and Portable Radio Research Group (MPRG), another research group of Wireless@VT [140]. OSSIE, like the GNU Radio and The Plastic Project of CTVR is currently using the USRP as the RF front-end.

6.2 CWT^2 SDR: Reconfigurable Waveform Framework

6.2.1 Platform Choice and Software Approach

To construct a complete CR node system, a supporting radio platform is necessary. First, it carries out all the radio operations required by the CE to fulfill waveform and network functionalities as a radio node. Second, it is the hardware testbed that verifies the functionality and evaluates the solution of the CE. To facilitate CE system design and expand its cognitive capability, the supporting radio platform needs to be both flexible and robust. To design such a radio from the beginning, an application oriented design approach is followed, as shown in Figure 6.1.

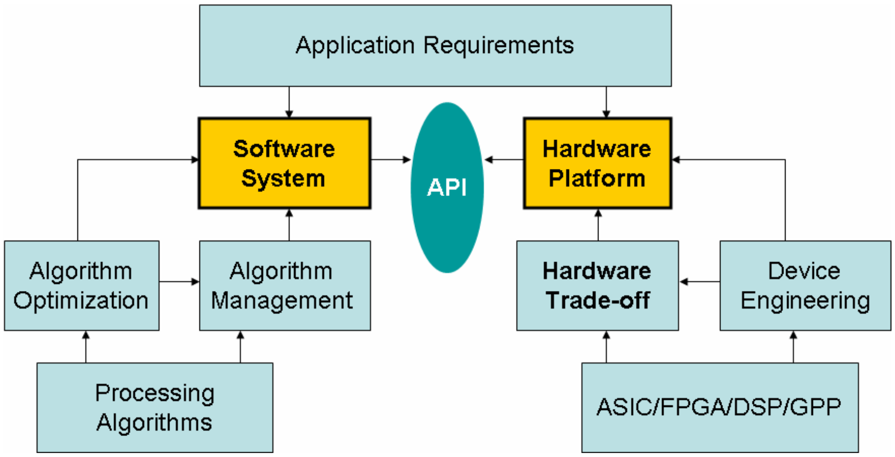


Figure 6.1: Application oriented design approach for CR platform

According the specific CR requirements stated in the previous section, there are several major system level requirements for this radio platform:

(i) Support maximal flexibility from PHY to NET (layer 1 ~ 3);

(ii) Provide general radio interface to the CE;

(iii) Provide maximal reconfigurability for runtime adaptability;

(iv) Open structure, easy to debug, maintain, and upgrade.

According to such requirements, the processing foundation is chosen to be the GPP with a standard Linux operating system; the radio software system is following objective oriented design using high-level language like C++ and Python; the signal processing structure is based on GNU Radio signal processing library; and the radio hardware front-end is USRP.

GPP with Linux OS

There are several reasons for picking GPP as the processor [141]. It offers a user-friendly compiler that supports high-level languages, thus increasing flexibility in software system structure design, such as component based and object oriented software engineering. Compared to other processors like FPGA or DSP, a GPP does not require additional specific knowledge and thus speeds the prototyping process. GPP eases the signal processing algorithm synthesis, thus it is suitable as a testbed solution for CR node system development. GPP is an ideal choice as the testbed solution for the complete control of both system level and modular level online adaptations, and the functional flexibility is fully supported by the reprogrammable code execution. GPP is seamlessly supported by a general Linux OS. With the open source software support the development environments are readily available. The Linux OS kernel is typically is POSIX-compliant, and therefore, an embedded structure can be created for system level testing from waveform to application. On the other hand, GPP is relatively slow, and its throughput is also limited by the system bus (such as PCI) and memory bus of the general computer structure. However, the current mid-to-high level of x86 computer hardware systems allow for a CR node to carry general narrowband waveforms with decent data rate, and this is enough for our public safety interoperability application needs (See Chapter 7.).

The operating system is important in system level design. It creates and maintains the working environment for all the software radio components across all communication layers. For software radio functionalities, the OS should provide such efficient resource management as memory efficiency and task prioritization to achieve real-time response.

Although an OS with a real-time kernel is preferred, the CWT^2 CR node system is built upon a general Linux kernel due to open source availability and development support. It is proven from system implementation benchmarking that given GPP processing power is sufficient for the computational requirements, general (non-real-time) Linux kernel can provide good real-time response for narrow band, low-to-middle data rates. This is because the GPP carries all the processing from both waveform generation and CE learning algorithms. When GPP is only working for the radio platform, several mega baud data rates are available and the bottleneck is shifted from processing power to the data bus, specifically the USB port for the USRP radio front-end.

Object oriented design is applied for radio software engineering. With the GPP-Linux environment, high level languages can be used to construct an open-structure, modular system. While there is a debate of C versus C++ to implement signal processing, the most critical parts of the signal processing chain takes less than 20% of the code [142]. They are mostly dealing with sample level processing such as IF filtering and channelization. Modular software structure offers significant gain in system level resource management and control in the other 80%, which could be overall beneficial. For our specific Public Safety Cognitive Radio (PSCR) system design which focuses on narrowband waveforms, processing power is not an issue, while the modular design advantage remains.

Universal Software Radio Peripheral (USRP)

The USRP is a software defined radio hardware device with a simple RF design that allows for a wide range of uses. It is completely open to the public, including the circuit schematic and FPGA source code. The USRP contains many different components, but the central hardware piece is the motherboard, shown in Figure 6.2.

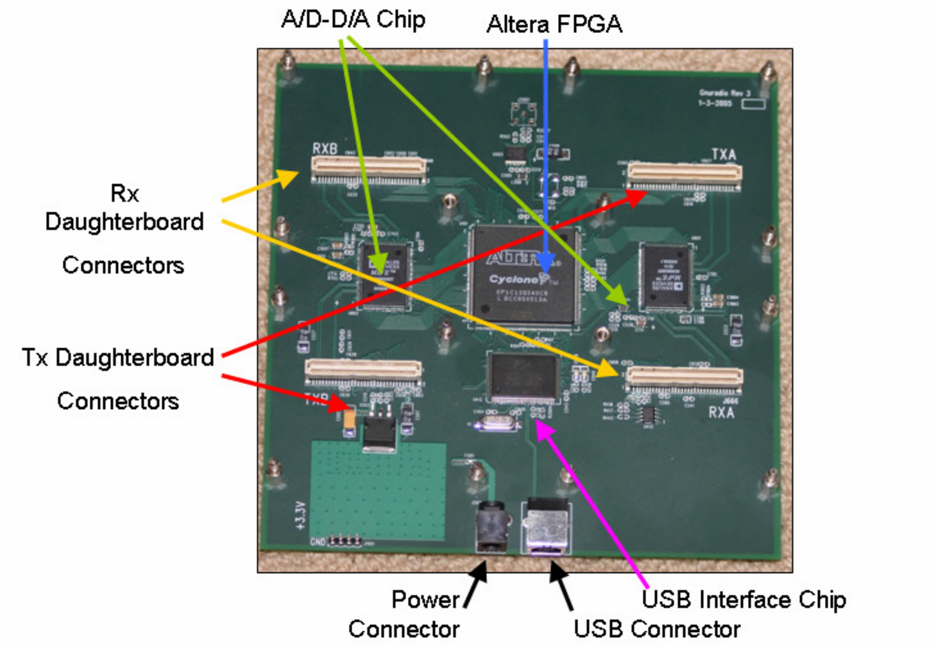


Figure 6.2: USRP motherboard layout

The USRP motherboard contains two AD9862 chips, each one incorporates a two-channel 12bit ADC and a two-channel 14bit DAC, an Altera Cyclone FPGA used for decimation, filtering, and up/down conversion, and a USB 2.0 interface for connection with a host computer. The motherboard does not have any filtering before the A/D converters, so a wide range of RF front-ends can be supported. Available RF daughterboards cover certain frequency bands, such as 400MHz, 900MHz, 2100MHz, etc., to allow for a variety of RF ranges to be supported. A list of current available daughterboards is given in Table 6.1.

Table 6.1: USRP daughterboard list with specifications

Board	**Rx / Tx**	**Frequency Range**	**Tx Power**
Basic Tx	Tx	100kHz – 44MHz	1mW
Basic Rx	Rx	100kHz – 44MHz	N/A
LFTX	Tx	DC – 30MHz	1mW
LFRX	Rx	DC – 30MHz	N/A

TVRX	Rx	50 – 870MHz	N/A
DBSRX	Rx	800 – 2400MHz	N/A
RFX400	Tx/Rx	400 – 500MHz	100mW
RFX900	Tx/Rx	800 – 1000MHz	200mW
RFX1200	Tx/Rx	1150 – 1450MHz	200mW
RFX1800	Tx/Rx	1500 – 2100MHz	100mW
RFX2400	Tx/Rx	2300 – 2900MHz	20mW

One single USRP motherboard can connect to up to two daughterboards; thus one board can support two different RF bands. Because the USRP has a USB 2.0 interface, it is compatible with almost all recently manufactured computers. The drivers for the USRP were primarily written to run in Linux OS, but some success on other operating systems has been reported. A lot of details of the USRP specifications are available from its designer's website [138].

To be used as the radio front-end to support flexible waveforms, the USRP's signal processing structure and capability are of great concern. See the general USRP structure shown in Figure 6.3.

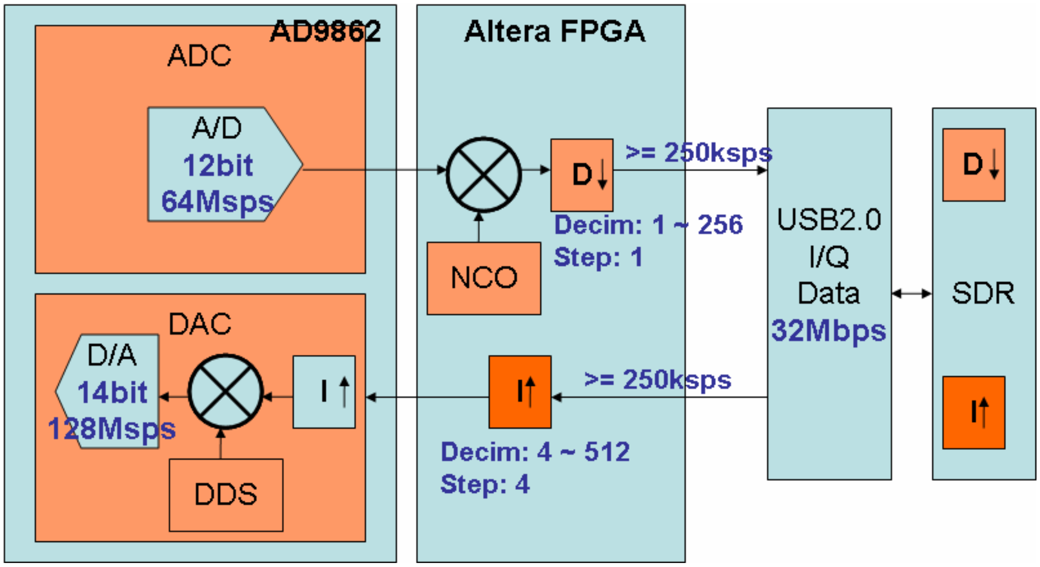


Figure 6.3: USRP signal processing structure

Along the receiving path, the output of the two-channel ADC is the series of samples of complex signal. Therefore, the digital down-conversion at FPGA is in the complex domain. With the current resolution limitation, the digital samples flowing to the computer via the USB interface should have a minimal sample rate of 250ksps. And, *vice versa*, the digital upstream to FPGA should also be at least 250ksps. This means that for low data rate waveforms, digital IF will have to be used. And further sample rate conversion and frequency conversion between IF and baseband is necessary in the digital signal processing by the computer. The USB2.0 interface provides 32Mbps throughput, which leads to a 4MHz complex signal bandwidth with floating-point (16bit/sample) resolution. However, such processing range is enough for the CWT^2 SDR application waveforms. The design challenge is not with processing power, but rather in the flexibility for reconfiguration and runtime adaptation, which will be detailed in the next section.

For the current FPGA configuration provided by the USRP, there is no real-time control over the FPGA. The FPGA gets programmed initially when the computer first tries to contact it. A chunk of FPGA code is loaded over the USB. When it is loaded up, commands are sent from the SDR in the computer to the specific registers in the FPGA that define tasks like the decimation ratio or center frequency, some of which are actually joint work between the FPGA and the DAC/ADC chips. Since the FPGA configuration is set once and kept static for all the following work, some important parameters that greatly affect the waveform flexibility, such as reference sample rate, initial channel bandwidth, IF tuning position, are difficult to change during the session. This imposes more challenges on the backend signal processing design in the computer.

There are two Programmable Gain Amplifiers (PGAs) connected to the output of the two DACs, which offers a 0 ~ 20 dB adjustable gain. The amplifier gain on the daughterboards is mostly fixed. Therefore, to extend the gain range, digital samples need to be scaled in their floating point format before they are fed to the USRP. The receiving Low Noise Amplifier (LNA) on some of the daughterboards provides several tens of dB gain. In the CWT^2 SDR system, the gain of the specific daughterboard used is calibrated and the gain is adjusted by changing the amplitude of the signal.

The USRP is designed as a stand-alone radio front-end that provides a standard set of device drivers as well as the FPGA API. It is used by many SDR platforms as described in a previous section. GNU Radio is currently the one of the primary software packages supported by the USRP. Hardware drivers for the USRP are included in the standard build of GNU Radio software package by default. Most of the USRP settings, such as center frequency, PGA gain, interpolation, decimation, and other transmit and receive path options on the USRP can be changed using GNU Radio. The drivers for the USRP in the GNU Radio package are provided both at elementary C++ class level and Python API function level. In the CWT^2 SDR design, the configuration and control related to USRP are mostly done by using Python API functions.

6.2.2 Signal Processing Structure

The USRP combined with normal computer creates a relatively inexpensive, very flexible hardware processing foundation for software radio testbed development, especially for academic research and system prototyping, which is exactly the purpose of our SDR platform design. With GPP, Linux kernel, and open source compiler, GNU Radio becomes a very handy base processing library to be used in constructing a software defined radio testbed for the CE. GNU Radio is briefly introduced in previous section. Although we will skip more general GNU Radio introductions (available from [34]), its processing structure and design trade-off needs to be addressed to raise the design issues of the CWT^2 SDR system.

The GNU Radio package follows object oriented software development using two object oriented languages, C++ and Python. Python is an open source, object oriented language, like C++, but has a scripting programming architecture. Therefore, Python code is more of a descriptive language, which is executed by instant interpretation instead of running after compilation. Python applies objective oriented principle to dynamic programming approach in order to achieve a balance between speed and flexibility [143].

Typically, the computational intensive tasks, like digital signal processing and system memory management, are coded in C++ and compiled as libraries; while transceiver level tasks, such as processing chain definition, and inter-module connections, are carried by Python. Building a signal processing flow using GNU Radio can be viewed as using Python as the glue to wrap and connect various C++ compiled processing modules to perform communication tasks. More specifically, GNU Radio uses another open source software packet, the Simplified Wrapper and Interface Generator (SWIG), to take C++ libraries and wrap them into a modular format that can be directly called by Python [144].

The job division between C++ and Python is a good balance between speed and flexibility. As detailed in the following paragraphs, the CWT^2 SDR software system development follows this combined-language objective oriented design approach. At the system level, there are mainly two issues of using GNU Radio package for SDR system design. The first is to use Python-C++ combined language advantages to implement flexible signal processing flow to generate real-time waveforms; the second is to use GNU Radio package provided libraries to create required signal processing blocks to generate layer 1, 2 reconfigurable waveforms. The design of the CWT^2 SDR platform, called "CWT^2 Waveform Framework", is basically to use GNU Radio tools to build needed signal processing modules, construct various signal modems and signal processing flow graphs that use the scheduler, and then, more importantly, use the Python object oriented threading mechanism to create a multi-threading waveform framework for runtime waveform reconfiguration and real-time link control.

Signal processing flow graph

Figure 6.4 shows a general software structure of using GNU Radio package in relation to programming threads (simultaneously running software tasks) for a transceiving session.

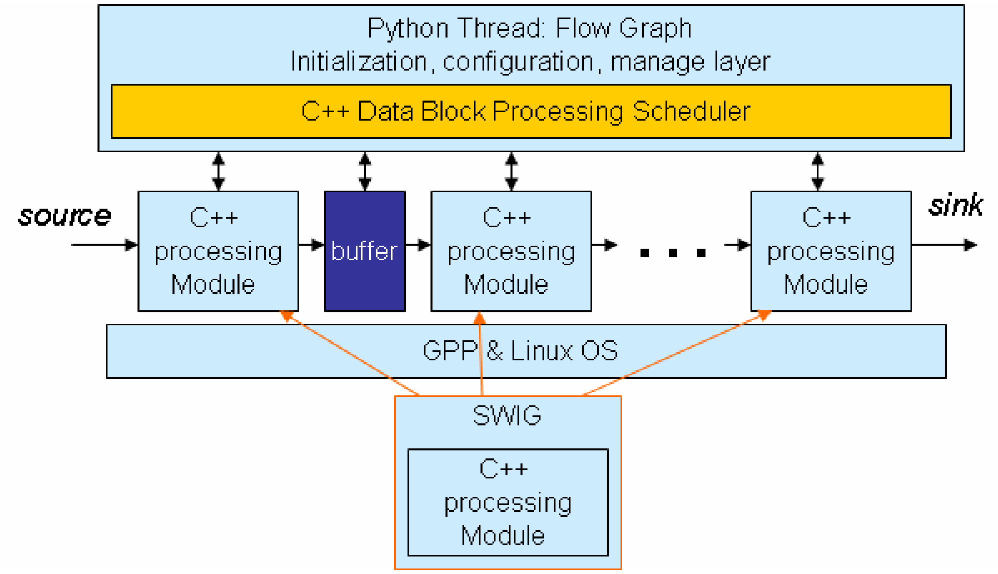


Figure 6.4: Software structure of GNU Radio based transceiving session

In GNU Radio, the C++ signal processing blocks are connected together by Python through the use of a processing chain structure. This processing chain structure is called a “flow graph”. Python defines a flow graph including a set of signal processing blocks, their interface specifications, and topological connection settings.

As shown in Figure 6.4, a flow graphs links together each source and sink pair as well as any intermediate blocks that are required to transfer the data stream from a source into a format that is understandable by a sink, for example, converting an FM radio signal that is received by a USRP (as the source) into an audio signal that can be played through a sound card (as the sink).

With such a flow graph defined and activated by Python, a C++ based flow graph scheduler is launched to define and allocate data buffers between different signal processing units and manages the consumption of these buffers for a continuous block based signal flow. A scheduler is associated with each active flow graph. Each scheduler is responsible for moving data through its flow graph. A scheduler iterates through the blocks in a flow graph, identifies blocks that have sufficient data on their input (s) and sufficient space on their output (s) to be able to process data and then triggers the processing function for those blocks. The scheduler in the current GNU Radio system relies on a steady stream of data input to the collection of blocks to cause the blocks to run and produce output. In GNU Radio, the signal processing flow is block based. The block buffers scheduled between signal processing units are monitored and managed in cycles, thus these processing units work in a "round robin" manner supervised by the flow graph scheduler.

In more detail, a flow graph runs through these steps:

- Python scripts wrap and connect signal processing modules (in C++) to build the signal flow graph
- Python calls the scheduler to check the integrity of the connections along the signal flow and verify the I/O data formats
- Python use the scheduler to define and allocate a common signal buffer memory space (with a unit of 32KB), which is shared by all the blocks in the signal flow graph.
- A multi-read-single-write buffer I/O control is applied for all the blocks sharing the same buffer space. Each block is designated (by the scheduler) with specific memory pointers for the input and output buffer.
- Then the signal graph is started; the blocks in the graph are working serially following the order of their positions in the signal flow. Say a graph consists of: A→B→C. First, A starts. A will check its input and output buffer status and starts to run when input (A's input is the source buffer) is available, and it stops when output buffer is filled. This also means that the input buffer of B is full. B will start running right after A stops, and keep running until its output buffer is full. Then C starts. When C stops with its output filled (C' s output is the sink buffer – the output of the whole graph), A starts to run again and the loop cycles until the input buffer is never refilled, which means that all the signal blocks have been processed. However, the flow graph is still active until it is terminated by external instruction from Python.

The flow graph makes the scheduling of the signal processing clear and straightforward. Although the signal processing tasks are carried by C++ libraries, the flow graph is under the total control of Python. By implementing threading methods in Python, various communication sessions can be generated, and with meta-threading control, flexible timing logics can be realized at the system level without changing the base signal processing libraries. Aside from session control, reconfigurability of the flow graph can be realized with standard interface interpretation methods, such as XML parsing, to support the platform independent radio interface described in Chapter 5.

The signal flow graph scheduler is directly incorporated in the CWT^2 waveform framework design. It is used as the primary real-time memory manager for various signal processing flows. Since it offers a standard API to be used in Python, the system design efforts can be primarily focused on block level and up instead of down to the C++ buffer pointer level. How the signal flow graph scheduler is implemented is detailed in the next section.

GNU Radio signal processing tools

GNU Radio is one of the Free Software Foundation projects. It is open source software, and is based on a lot of other open source libraries. As stated before, GNU Radio requires SWIG to bridge C++ library and Python objects together. Also, it uses a C subroutine library Fast Fourier Transform in the West (FFTW) [145] to process discrete Fourier transforms; it uses Linux USB libraries to access the USB2.0 interface of the USRP radio front-end; it uses universal Advanced Linux Sound Architecture (ALSA) [146] driver to provide audio/voice service in different computer hardware settings. With the support from these and a lot of other open source software tools, GNU Radio combines a powerful set of signal processing tools that can be easily accessed in Python and C++ environment. They can be directly used to run simple calculations such as a block spectrum display or telephone dial-tone generation, but more importantly, they can be used to create new processing blocks and even complete flow graphs to realize a signal receive or transmit task. The latter is how GNU Radio tools are used in the CWT^2 SDR system signal processing. More specifically, a set of important tools used is listed below.

(i) FIR filter designer

The most powerful toolset offered by GNU Radio is the FIR filter designer. It provides various filter design methods, either in time or frequency domain. It also provides many window types for ripple conditioning. It has been extensively used in making real and complex adaptive filters in CWT^2 signal processing blocks.

(ii) Rate conversion

Together with filters, digital sample rate conversion is one of the key issues in reconfigurable signal processing system design to support multiple waveform formats. The sample rates at different stages of the signal processing chain should be determined to synchronize the working speed between the processing blocks before connecting them. In GNU Radio, interpolation and decimation are integrated with filters; different interpolating curves and filter types can be combined to optimize the linearity and spectrum.

(iii) FFT calculation and graphic display

FFTW is directly called in GNU Radio to calculate signal spectrum in block based processing. The real-time processing throughput depends on block size and thus on the frequency resolution used with a specific bandwidth. One handy tool GNU Radio provides is a graphic display panel written in wxPython, which is a Python add-on toolkit for plotting in Linux X-windows [147]. However, using wxPython for real-time spectrum display costs significant computational power. Thus it is largely used for low data rate signal spectrum display in CWT^2 SDR system, such as for showing public safety waveform spectra in our Smart Receiver Demo at SDRF Technical Conference 2006 (See Section 3.9.1.). For high data rate links, link statistics are more commonly used as performance monitoring methods.

(iv) Standard signal sources and mathematical operations

Standard signals sources provide sinusoidal signals, Gaussian noise, pseudo-random sequences, constant values, etc. These are available in both real and complex formats. Standard operations that are widely used includes add, subtract, multiply and divide. There are also complex operands like arctangent and absolute-value. With these basic elements a lot of important processing blocks are built in the CWT^2 SDR system, such as frequency up-/down-converter, Numerical-Controlled Oscillator (NCO), phase detector, etc.

(v) File interface

GNU Radio signal processing is fully C++ based; as a result, the low-level data interface is defined in C++. Standard ASCII files and binary files are used to store processing data, and the C++ file handler is wrapped in Python API. Therefore, these data files can be directly created as normal signal processing blocks and connected with other normal signal processing blocks in the signal flow graph. The Python file interface makes it a lot easier for the designer to focus more on waveform design, as in the CWT^2 SDR system. Data files are fully managed at the Python level for on-line status logging, data saving, off-line debugging, and post-processing. For example, in signal recognition, first-order signal statistics can be calculated in real-time, while the data is saved for second-order spectrum or other computation intensive calculations running in the background. The recognition results are encoded into radio environment XML file saved for any module that needs this information.

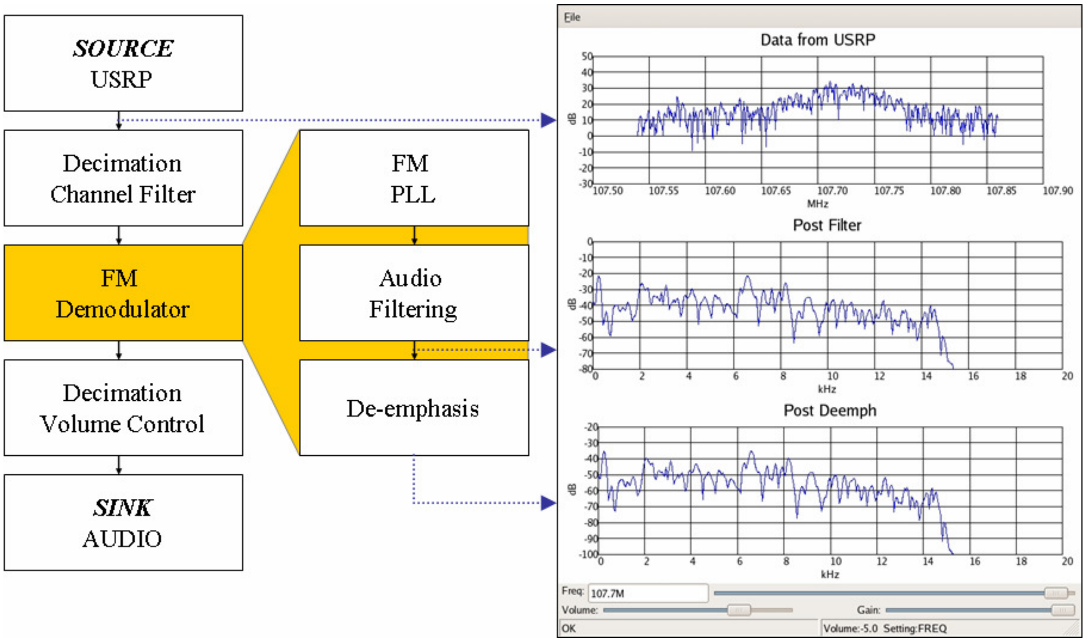
(vi) Audio interface

Both Linux sound card drivers, Open Sound System (OSS) [148] and Advanced Linux Sound Architecture (ALSA) [146], are supported in GNU Radio. ALSA is more popular and more compatible with modern computer hardware systems, thus is picked by the CWT^2 SDR system to provide voice and audio services. Similar to file interface, GNU Radio also wraps ALSA in the form of Python signal processing blocks for easy use.

(vii) USRP API

Although USRP, as an independent radio front-end, provides standard APIs, GNU Radio wraps these APIs into Python modules. Important USRP hardware functions like NCO tuning, digital frequency conversion, sample rate conversion, A/D interface sample rate, and more RF settings in the daughterboard, can be fully controlled via Python blocks. In this way, the key PHY parameters on radio front-end can be easily managed by Python flow graph for waveform reconfiguration.

To illustrate how these tools are used, Figure 6.5 shows a simple FM reception signal flow graph. There are mainly five processing blocks. The source block is the USRP providing RF signal down-conversion to digital baseband, and the sink block is the audio device that plays the demodulated audio signal. First stage sample decimation is done between the USRP ADC and the FM demodulator, in order to save computational cost for demodulation. Second stage sample decimation is done between demodulator and audio device, in order to meet the device compatible sample rate for correct audio reply. In the middle, a digital PLL is followed by an audio filter and de-emphasizer for FM demodulation. The spectrum display on the right is carried by wxPython. Flow graphs like this one are created in the CWT^2 waveform framework. They are configured and controlled in real-time for different waveform and link requirements.



Freq: 107.7M

Volume:

Gain:

OK

Volume:-5.0 Setting:FREQ

Figure 6.5: FM signal receive flow graph with spectrum display

6.2.3 CWT^2 SDR Waveform Framework Structure

The CWT^2 waveform framework system is structured with two planes, a control plane and a work plane. The control plane is on top of the work plane. To support cognitive radio functionalities, the control plane should support run-time waveform reconfiguration and session control, and the work plane should provide real-time adaptive signal processing according to the control plane instructions.

For waveform configuration, each plane has a hierarchical structure to carry on the configuration from the cognitive engine down to the lowest level of signal processing units, as shown in Figure 6.6.

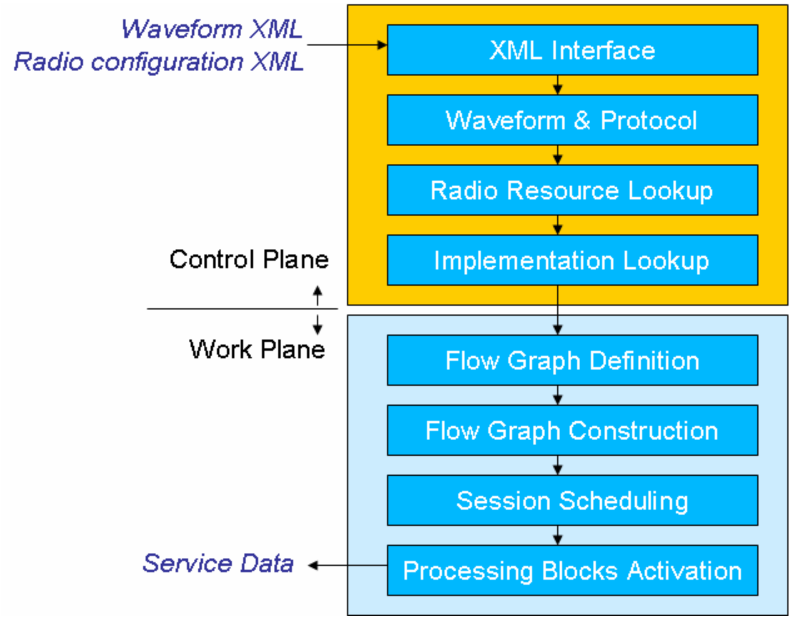


Figure 6.6: Hierarchical two-plane structure of the CWT^2 SDR system

As stated in Chapter 5, XML is used to describe both the waveform and radio configuration profiles to the radio. The general radio interface at the CWT^2 waveform framework provides an adaptive parsing interface to translate XML information into a tree structure in the memory, as shown in Figure 6.7. The XML parser translates the XML contents into standard Python objects. As shown in Figure 6.7, there are two root nodes in this object; one is the root for radio platform configurations, and the other is the root for waveform parameters. More detailed description and a user manual for the CWT^2 SDR general XML parser are included in the PSCR node system package release version 1.0. See details in Appendix B.

In the control plane, once the configuration tree is extracted from XML files, it is mapped to specific radio resources and processing implementations. All the needed processing resources are called to be a processing block in the flow graph, and their working relations are clarified to define the connection and I/O data format in the flow graph. Then the work plane takes over and a flow graph is defined according to such information. Once the flow graph is verified and buffers are allocated, the flow graph scheduler will simply start the signal processing flow.

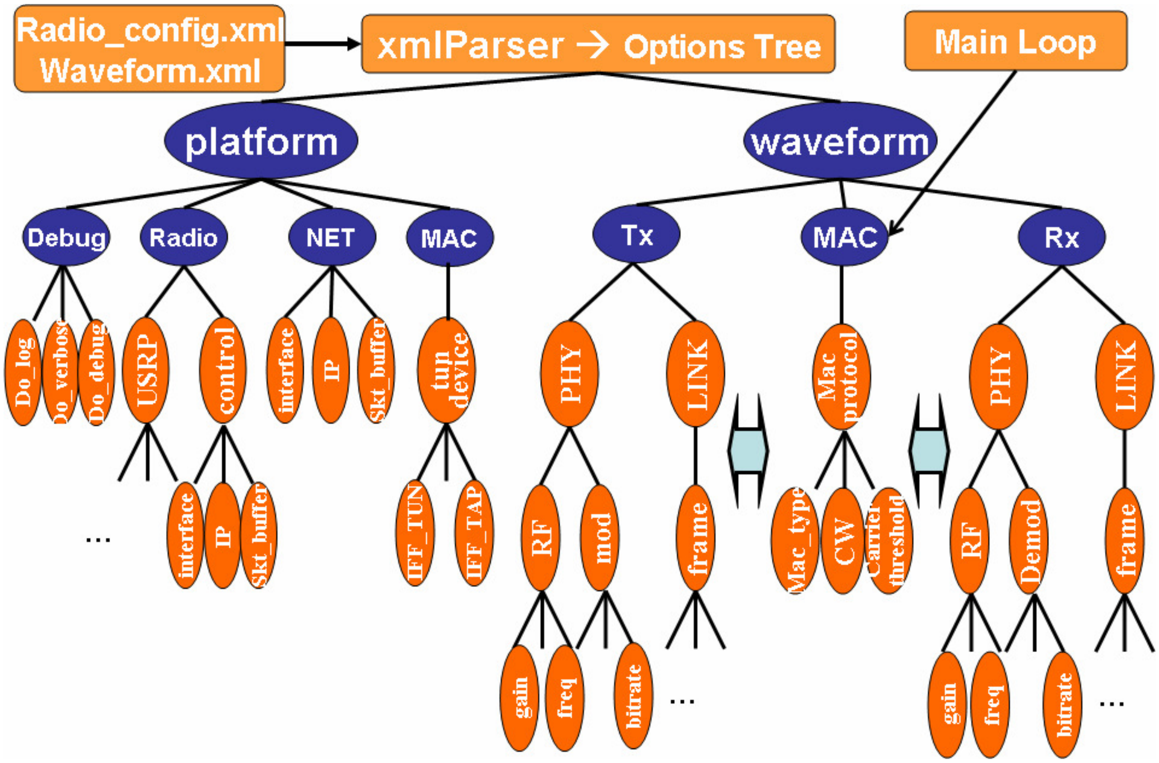


Figure 6.7: CWT^2 waveform framework XML parsing memory tree

Although this seems simple to go through a straightforward waveform configuration, there are a lot of hidden design challenges, such as using one configuration structure to support various waveforms with different parameters across PHY and MAC layers, verifying and carrying on corresponding radio actions on the fly. Moreover, a radio should be much more mature than a simple signal processing kick-off. Supporting waveform reconfiguration indicates a flexible control mechanism is necessary to carry various link behaviours according to the waveform protocol, and such control mechanism should be at runtime for radio applications. For example, an analog voice service waveform may need a push-to-talk access control, while a digital video streaming needs a standard TCP/IP packet flow via a carrier sense collision avoidance type of MAC protocol. For the latter case, when the configuration Python object is loaded with the name of the protocol, the radio needs to define and initialize a specific algorithm to control the signal processing flow, the flow graph. Since the flow graph is running in a real-time session managed by the flow graph scheduler, which is a C++ thread [149], the controlling algorithm needs to be another thread to control the scheduler thread. Therefore, a multi-threading control system is needed in CWT^2 waveform framework system for both on-line waveform reconfiguration and real-time link control.

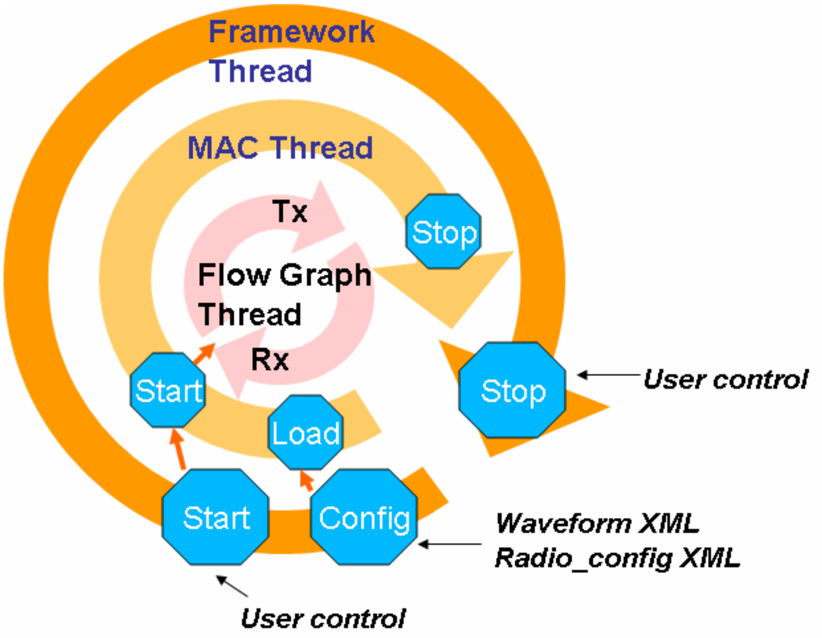


Figure 6.8: CWT^2 waveform framework multi-threading control

The multi-threading control in the CWT^2 waveform framework is shown in Figure 6.8. It also has a hierarchical structure that consists of three layers of threads. The system top level thread is the framework thread that interfaces configuration and waveform level service session control. It loads the specifically required MAC into the second level thread and manages its states. The MAC thread further controls the third level thread, the signal processing thread, by the flow graph scheduler. The flow graph scheduler C++ thread also directly calls the Linux OS API, which is real-time. The outer two levels of threads that take care of waveform configuration and link control at runtime are implemented in a standard Python threading object, which is directly calling the Linux OS threading API, too. Therefore, although the threading is created as Python object, it is directly handled by the POSIX OS kernel. Thus the system overhead is minimized, and a relatively good real-time response is achieved. A complete waveform and link switch in the CWT^2 waveform framework only takes less than 0.1 second, and, if only some of the PHY parameters are need to be changed, the time is negligible.

Linux OS kernel provides the multi-threading processing capability, and the control between the threads at different levels is implemented through the use of standard OS threading events. Therefore, the interaction between C++ thread and Python thread is seamless and efficient. It fully exploits the GNU Radio's advantageous design trade-off between C++ and Python. With a good balance between waveform flexibility and real-time response, it offers a mature radio platform that is not limited to a single transmit or receive task, but rather a reconfigurable waveform solution covering PHY, MAC and NET layer specifications. By feeding standard platform-independent waveform and configuration XML files, the CWT^2 waveform framework directly provides service by coordinating its processing resources to generate the required waveform and service, as detailed in the next section.

To provide network based distributed cognitive radio functionality, the control plane in CWT^2 waveform framework is implemented using standard TCP sockets to provide a universal interface for both local node and network intelligence. Since the framework is in Python, the TCP control sockets are realized as standard Python objects. In controlling the waveform processing, each configuration/control thread in the waveform framework has an independent control socket. Each control socket is designated by a specific control port number, and can be bound to any standard network interface. For example, when the cognitive engine is at the same CR node with the waveform framework, all the control sockets can be bound to "localhost" virtual network interface provided by OS. Then the configuration and control can be transferred to radio platform via local looping. When the cognitive engine is at a remote side, the sockets can be bound to standard network interface with an accessible IP address. And the configuration and control information can be delivered via standard TCP/IP tunnel. Such a network interface can be the digital link through air interface created by the radio platform, and the radio adapts its PHY and MAC functionality according to the received cognitive instructions through the adaptive link. By providing flexible network realization of the general radio interface, a distributed cognitive radio network can be formed, which exactly matches the network based vision of cognitive radio functional structure (See Section 2.1.3).

Figure 6.9 shows the working mechanism of the standard TCP sockets. The socket is configured and initiated by the configuration and control threads. The data reception is triggered by the listening event on the specific port. The control data typically includes link behavior instructions such as start, stop, pause, mute, etc., and the configuration data mainly includes waveform and radio configuration. The information delivered from cognitive engine to the radio platform can be as complete as a full solution, as simple as a transmit/receive timing.

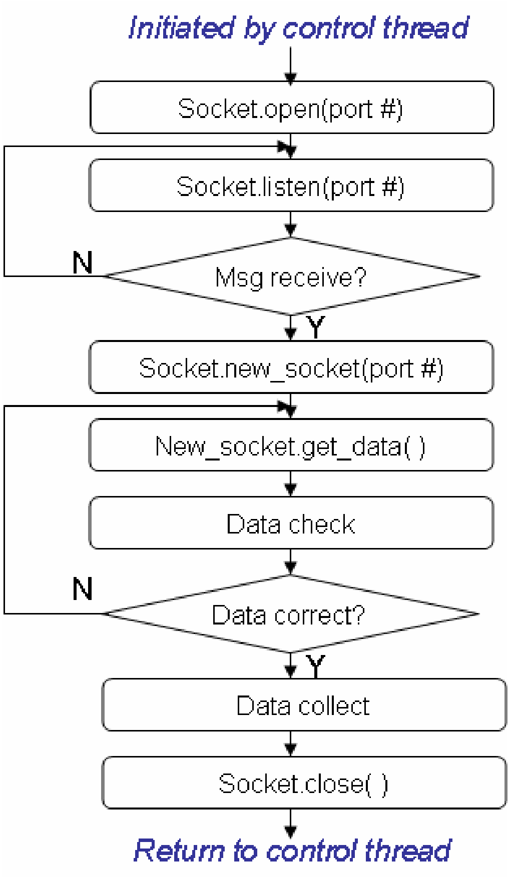


Figure 6.9: Working chart of the TCP socket for radio control threads

6.2.4 CWT^2 SDR Waveform Development

As shown in Figure 6.6, the work plane in CWT^2 waveform framework should generate different waveforms and support the adaptation required from the control plane. For a specific waveform, a set of implementation-level processing resource needed to generate this waveform should be claimed and verified to be ready to work. Then the settings of these resources should be defined to match the specific waveform requirements. Some service related radio sources are also included, such as audio interface or digital network interface. In CWT^2 waveform framework, such resource management process is implemented in database lookup, as shown in Figure 6.10.

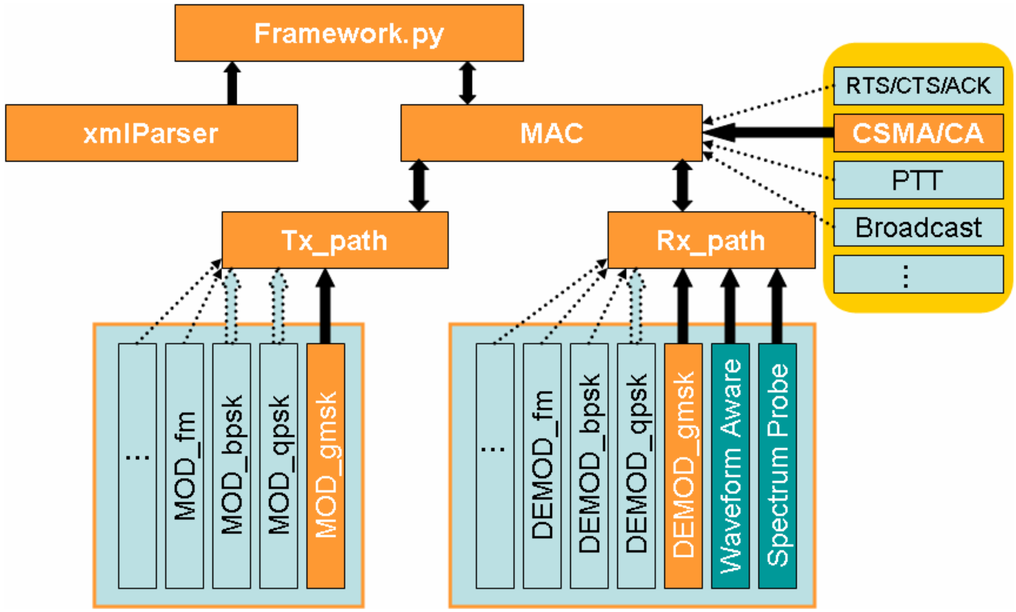


Figure 6.10: PHY-MAC reconfigurable waveform implementation management

The CWT^2 waveform framework is a purely modular structure. PHY-layer baseband specifications for different waveforms are modularized into different modems. Each baseband modem takes care of one group of modulations, and can be reconfigured for a large operational range. For example, an FM modem takes care of all analog frequency modulation waveform with any modulating index and channel bandwidth; and a BPSK modem can support binary keyed modulation with different data rate, pulse shape, differential symbol mapping, etc. Such a modular PHY structure is specifically designed for two purposes. First, this provides the best support for a general radio interface. With a waveform configuration input at the communication level, a translation to platform implementation settings is a must. The modular lookup makes radio resource management straightforward and efficient. Such a lookup can be easily updated when processing capability is changed, in the format of supported baseband modem with its parametric operational range. Second, it is designed to provide the exactly support for waveform recognition capability needed by the CE. Specifically, this is the two-stage signal classification module that needs a modem to support a group of modulations. For BPSK modulations with different symbol rates and pulse shapings, the CWT^2 waveform framework uses the same modem with only a few parameter setting changes. Also, in the CWT^2 waveform framework, there are two additional modules, spectrum sweep and waveform aware. They specifically support radio environment recognition functions for the cognitive radio system.

Besides the modular PHY layer, the MAC layer is also modularized. Due to the multi-threading control plane design, the MAC and PHY can be independently configured and controlled at different levels of threads, as shown in Figure 6.8. Therefore, both the PHY and MAC can be combined as "plug-and-play" to generate a complete waveform. Of course in a real link deployment, certain MACs are typically used with certain PHY specifications, especially when the application requires a set of defined waveforms. The designed waveform framework structure accepts a free combination of any two modules from PHY and MAC to carry purely user-defined waveforms, which enables the "cognitive" waveform support for cognitive radio nodes.

In the work plane, when specific waveform implementation "sheet" is received from the control plane, a specific instance of a waveform framework is created in software, as shown in both Figure 6.11 and Figure 6.12.

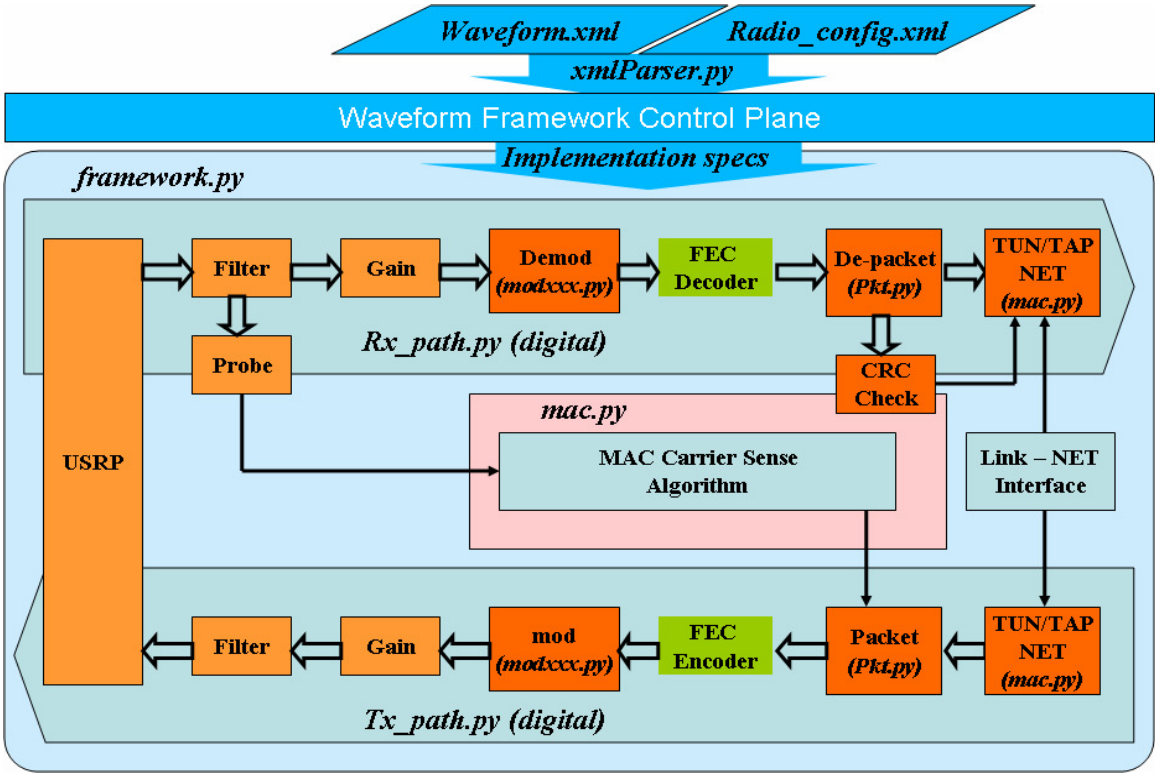


Figure 6.11: Example digital waveform generation in waveform framework

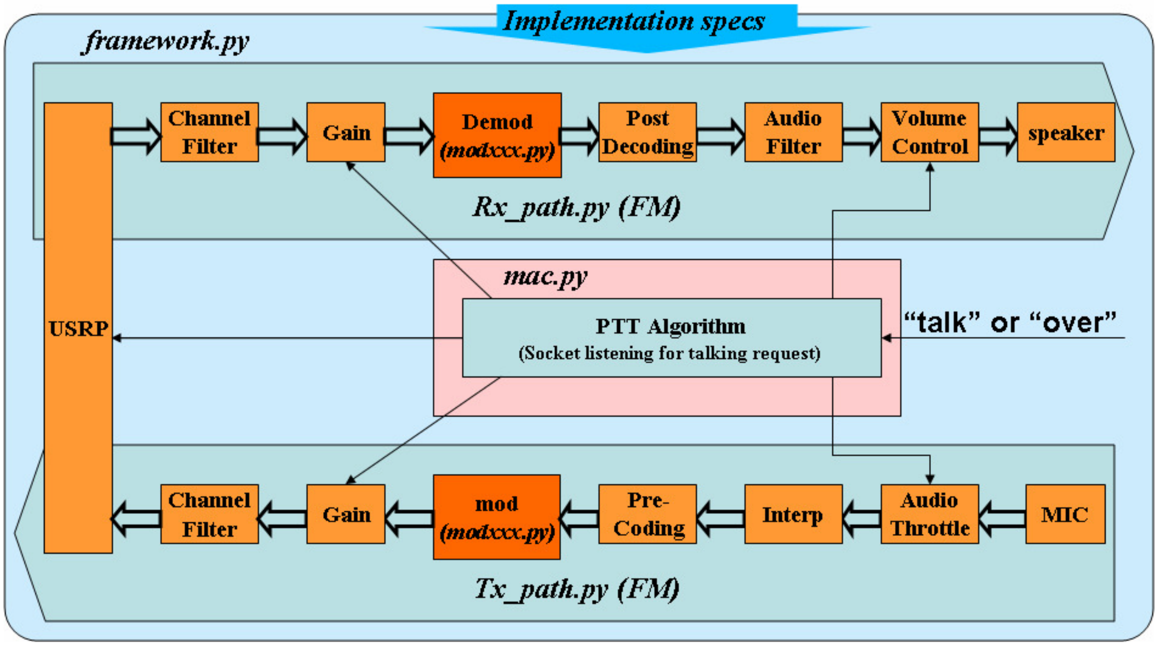


Figure 6.12: Example analog waveform generation in waveform framework

A general digital waveform is generated in Figure 6.11. The PHY can be a typical digital modem such as PSK or MSK, and a carrier sense MAC is loaded to control the link. The signal flow graph consists of both transmit and receive signal paths. The error correction blocks are currently not implemented. Note that for the digital link, a virtual network interface is created with TUN/TAP open source library [150]. This software created virtual network interface provides a universal standard service interface for all kinds of digital applications, such as email, messaging, voice over IP, and video streaming. Figure 6.12 shows another example of analog FM waveform generation. The signal flow graph also contains transmit and receive paths. Note that the service interface resources are changed to microphone and speaker. And a simple Push-to-Talk (PTT) algorithm is loaded as the MAC to control the voice link.

In the waveform framework work plane, all the link control, processing resources, and signal processing are automatically created and managed. The complete waveform framework system, including control and work planes, is implemented in Python/C++. All the processing tasks are carried by the single GPP in the computer. Note that in the current system design there's no special mechanism to guarantee the real-time response. With the limitation of the general (non-real-time) Linux OS kernel used, the "real-time" performance is achieved with excessive processing power. When processing power is not enough, buffer depletion and thus a performance discontinuity will occur. For now there are two continuing development tasks remaining for the CWT^2 waveform framework. One is to port this SDR to a multi-processor hardware platform such as the Cell Broadband Engine to claim more processing power; the other is to translate all Python based multi-threading control mechanisms into compiled C++ objects to reduce processing cost.

6.2.5 Digital Signal Processing Design Issues

A detailed general description of how to use GNU Radio provided signal processing blocks is provided by our report [129]. In the CWT^2 waveform framework, the radio front-end is configured to directly down-convert the RF signal to baseband. The waveform signal processing in the waveform flow graph starts at the digital quasi-baseband where the signal contains carrier offset and is not synchronized. The input samples are in complex format. Due to this advantage, quadrature signal processing tasks in carrier synchronization and symbol timing are fully implemented in the form of real and imaginary parts, thus the I/Q branches are integrated into one complex signal flow. More importantly, since the signal is in complex format, the down-converted signal can be directly centered around zero frequency instead of having a relatively high carrier frequency to avoid image overlapping. Therefore, the sample rate is greatly reduced and the computational cost is minimized (See Chapter 3).

In the flow graph implementation, the signal processing blocks in the flow graph use complex multiplication, complex filtering, and complex phase differentiation. And the signal data formats between processing blocks are typically inherited from a common complex vector class. In the flow graph signal processing, the data object inherent methods are massively used, thus computation efficiency is gained since there is no need to search the entire memory and call standard external methods for a basic calculation every time. This is basically trading a relatively small amount of extra memory for a big speed increase, which is suitable for real-time processing. The complex vector object oriented signal processing is also applied for signal classification module in the CE, which is detailed in the CWT^2 PSCR node system package user manual, which is listed in Appendix B.

Sample rate conversion is one of the key design issues for waveform flexibility. Typically a digital sample stream with 4 ~ 6 samples per symbol is used for sample level processing tasks such as carrier recovery and phase lock. It is also used for signal classification tasks. Therefore, for narrowband signals a decimator is needed following the output of the USRP FPGA which always has a sample rate over 250ksps due to its resolution limit. The decimation ratio is adapted according to the specific waveform's symbol rate, pulse shaping roll-off and modulation scheme. In symbol timing, the sample stream is typically interpolated from 8 to 10 samples per symbol for the pulse reconstruction filter to reach a satisfying trade-off between computational cost and timing accuracy. The output of the symbol timing block is the decision statistic values at symbol rate. For analog waveforms, there is no symbol timing. And to minimize waveform distortion in analog modems, the sample stream is decimated to around a couple of hundred kHz after the FPGA. Once the baseband (audio) signal is demodulated after the modem, it is further decimated to meet the required audio bandwidth to save computation. It is also set to be compatible with the audio device in the computer. A simple analog flow graph is illustrated in Figure 6.5, and a digital BPSK flow graph is illustrated in Figure 6.13.

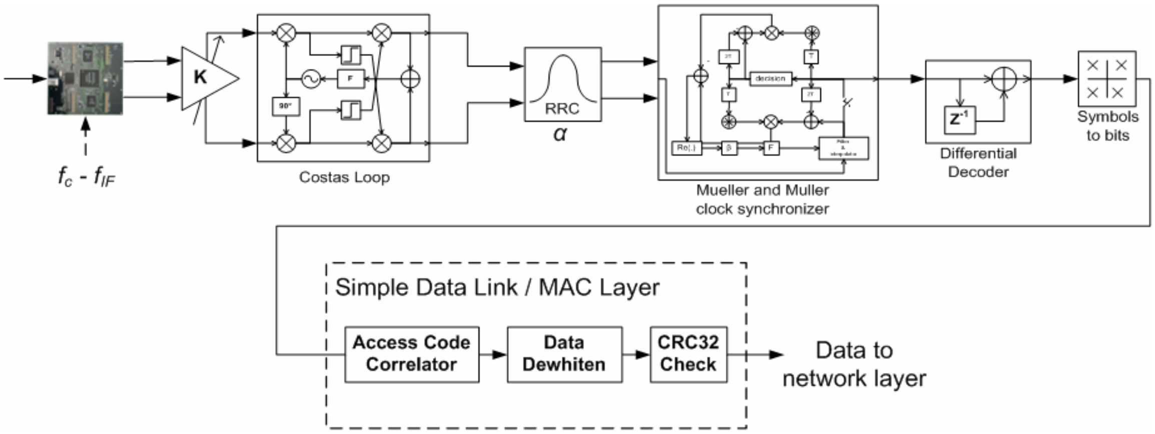


Figure 6.13: Example differential BPSK signal flow graph

6.4 Summary

The software based radio design approach reduces the content of analog components and provides the highest level of waveform flexibility and reconfigurability. Thus SDR is best preferred as the supporting radio platform for the adaptive and cognitive radio and networking functionalities. The CR system imposes strict requirements in its supporting platform design to fulfill cognitive functionalities as well as normal links. Generally, awareness and adaptation needs wideband sensitivity and linearity in the radio front-end. They also require real-time reconfigurability and waveform flexibility in signal processing and link control. The design trade-off is jointly set by cost budget and target application scope.

For CWT^2 CR nodes, a SDR platform called "waveform framework" is developed that provides waveforms fully reconfigurable at the PHY and MAC layers. It uses GPP and Linux as a processing foundation for high level reconfigurability and a friendly development environment; it uses Python and C++ languages to achieve a balance between flexibility and real-time response. The waveform framework uses GNU Radio software package to realize real-time signal processing structure, and it uses a USRP as the RF front-end. Its system architecture consists of control plane and work plane. The control plane handles the general radio configuration and control interface with the CE; converts the received general waveform solution into platform specific processing resource coordination; and launch a multi-thread waveform processing control to generate the specific waveform solution. The working plane follows the processing resource coordination, constructs a real time signal processing flow for the specific waveform, and runs under those control threads from the control plane.

Various signal processing and service modules are developed in the CWT^2 waveform framework platform to support various required waveforms and services. Signal data is defined as a complex vector object to facilitate real-time signal processing.

Chapter 7: Public Safety Cognitive Radio Node

7.1 Complete Cognitive Radio Node System Integration

With all the functional subsystems designed, now a completed cognitive radio node can be integrated, as shown in Figure 7.1.

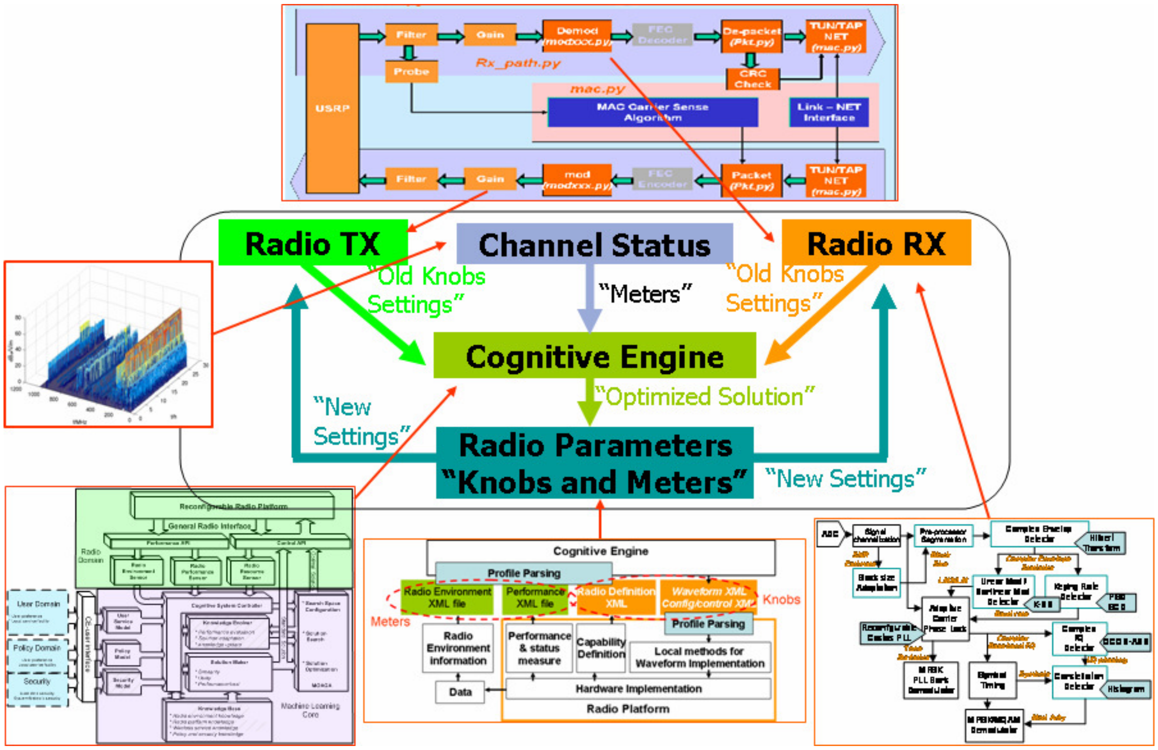


Figure 7.1: Cognitive radio node system integration

Figure 7.1 illustrates how the concept of the cognitive engine finally leads to a complete cognitive radio system. The block diagram at the center shows the definition of the CE, "reads the radio meters, makes a solution, then turns the radio knobs". The radio shown on top is the CWT^2 waveform framework (Chapter 6); a platform independent interface is created between the cognitive algorithms and the radio platform (Chapter 5); the radio environment awareness is designed and implemented in C++ (Chapter 3); the solution making and learning is realized with CBR-GA chain (Chapter 4). A detailed system architectural view with all key functionalities and interfaces is provided by Figure 7.2.

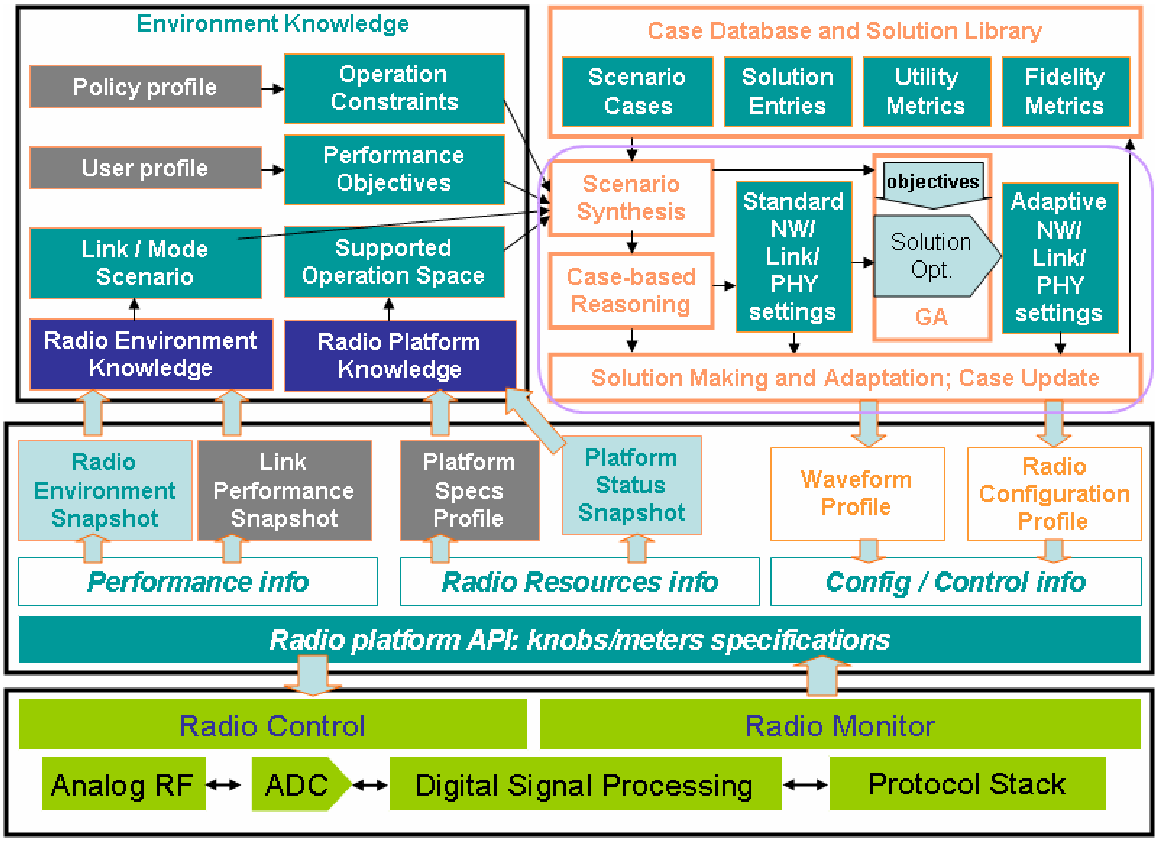


Figure 7.2: Detailed cognitive radio system architecture

All the blocks and interfaces are already explained in previous chapters. This CR architecture forms a general node solution for various cognitive applications. Due to its network based standard interface design (See Chapter 6 and Section 7.2.), the included functional modules can be spread over the network for distributed cognition.

Based on this architecture, a specific fully-functional cognitive radio node system is implemented for public safety interoperable communications, which is the focus of this chapter.

7.2 Public Safety Cognitive Radio Node Implementation

Today, more than 55,000 separate public safety agencies operations are based on disparate technologies, frequency bands and protocols. The U.S. Conference of Mayors published a survey of interoperability in June 2004 documenting interoperability challenges facing public safety. 88% reported that they are not interoperable with Homeland Security; 83% reported that they are not interoperable with the Department of Justice; 94% reported that they do not have interoperable capability between the police, fire, and emergency medical service [151].

The Department of Homeland Security defines interoperable communications as "the ability to communicate and share information as authorized when it is needed, where it is needed, and in a mode or form that allows the practitioners to effectively use it." [152] Public safety agencies require interoperable voice and data communications that can support coordinated responses to incidents ranging from a major terrorist attack such as 9/11/2001 to the 1993 standoff in Waco, Texas, to the 1996 rural Montana arrest of the Unabomber, to local fire and law enforcement operations.

In an ideal world all public safety personnel would carry radios that are compatible in waveforms, protocols, and frequency coverage, and - subject to limitations imposed by the incident command structure - they could take advantage of full interoperability as defined above to form and interconnect networks as any situation required. Spectrum assignments would be flexible while ensuring immediate priority access by public safety agencies for emergency use. Meeting these needs under real world economic, political, and commercial realities requires the development of affordable radios with (a) sufficient frequency and waveform agility to interoperate with any of the existing waveform formats, (b) the capability to serve as a gateway between two or more networks using incompatible modulations, and (c) the intelligence to find spectrum, configure themselves, and begin operation in response to simple operator instructions subject to the incident chain of command.

SDRs can typically offer (a) and (b), since their operating frequencies and waveforms are software controlled, and switching between modulations and protocols simply requires running different code. Cognitive radios (CR) add the intelligence needed in (c) on top of the SDR. Cognitive radios are aware of their environment and intelligently adapt their performance to the user's needs with its cognitive engine (CE). The CE responds to the operator's commands by providing suitable solutions configuring the radio for whatever combinations of waveform, protocol, operating frequency, and networking are required. It monitors the radio's performance and adjusts its settings to deliver the needed quality of service subject to the combination of user requirements, operational limitations, and regulatory constraints. The CR does this with minimal operator involvement and will require little or no retraining of personnel. For PS interoperability the cognitive radio could interoperate with everything that is out there. It can work as a gateway interconnecting mutually incompatible networks, allowing full interoperability between users with different radios.

From summer 2005, CWT was awarded a Communications Technologies (CommTech) project by the National Institute of Justice (NIJ) to apply cognitive radio technology for public safety interoperability [153]. The target application requirements are decomposed into three non-overlapping use cases to formulate the PSCR system design.

(i) use case 1: full interoperability between various incompatible PS waveforms
(ii) use case 2: cognitive radio environment sensing and link/network behaviour
(iii) use case 3: fast system implementation and field deployment

Use case 1 is to provide the universal communication service of information exchange of group voice traffic, digital data, position, etc. The PSCR should be able to serve as a gateway to bridge incompatible waveforms and networks, or as a multi-mode multi-band wireless terminal for the user. Use case 2 needs the PSCR to be able to sense the frequency band of interest, detect and identify existing PS waveforms and networks, and report to the user to enable the awareness of the radio environment. Use case 3 is demanding the PSCR system, especially the cognitive algorithms to be easily cooperated with various radio platforms, and to be fast deployable on the field with little prior environment knowledge.

Based on the general cognitive radio architecture, a Public Safety Cognitive Radio (PSCR) node system is designed to fit the above use cases.

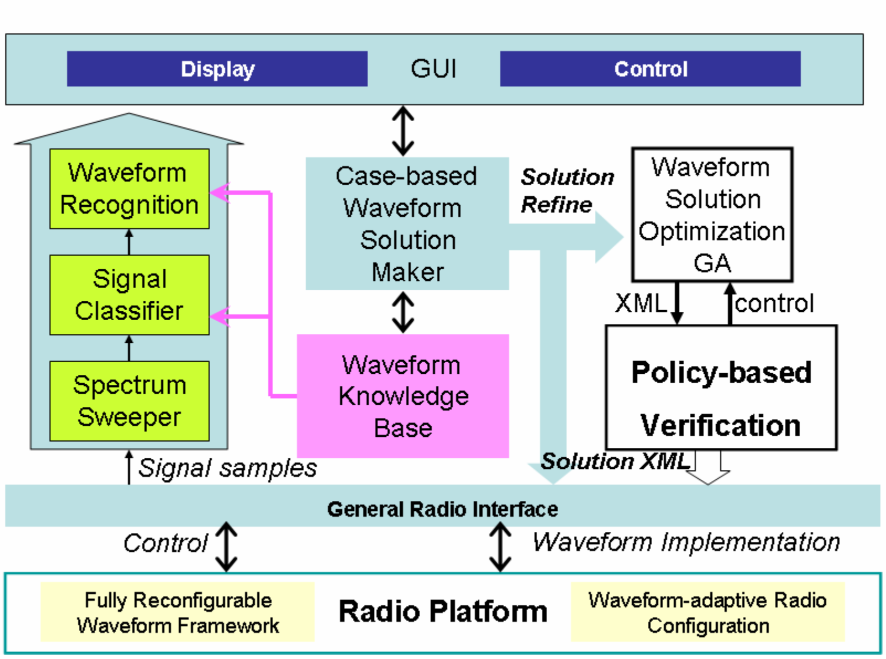


Figure 7.3: PSCR node system block diagram

Figure 7.3 gives a system overview of the PSCR node. It consists of four major subsystems. Among which the kernel part is the CE that is implemented as an application specific version of the CE in the general CR node (see Figure 7.2) specifically for PS application use cases. The major difference is that in PSCR CE the solution making module is simplified to be case based reasoning (CBR) only (See Chapter 4 for details of CBR solution making), because PS communications use pre-defined waveforms, and thus customized waveforms are not needed from a GA solution search. However, a GA may still be needed to improve the link performance by adjusting parameters of these PS waveforms. The GA solution search module is switched off in current PSCR node. The second subsystem is the Graphical User Interface (GUI); the third system is the CR knowledge base containing PS domain knowledge to support solution making; and the fourth subsystem is the radio platform for PS waveforms. A detailed implementation view of PSCR node is shown in Figure 7.4.

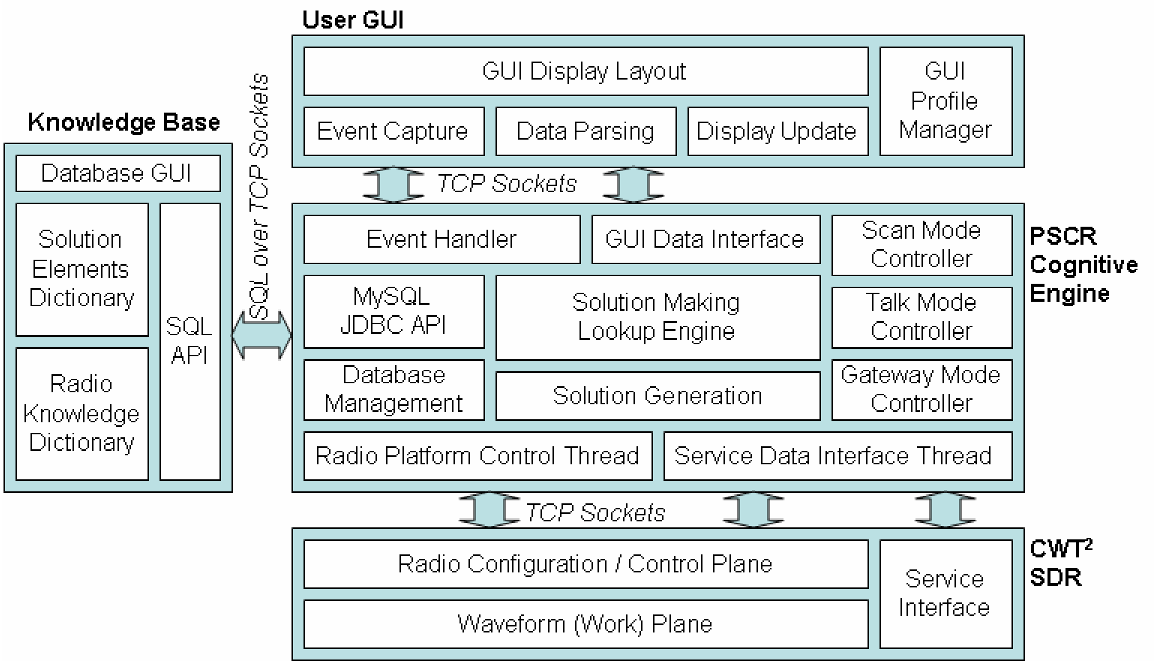


Figure 7.4: PSCR node with subsystem implementation details

To support use case 1 and 2, three working modes are defined for the PSCR node. Scan mode is used for the PSCR to gain radio environment awareness. As shown in Figure 7.3, a three-step recognition procedure is designed. The first step is spectrum sweep, and thus energy at channels of interest is picked; the second step is signal classification and thus modulation and other signal parameters are classified; the third step uses the results from previous two steps and consults the waveform data base to obtain full knowledge of the recognized PS waveform. With such knowledge a link can be established via this waveform if needed. Talk mode is used for the PSCR to establish a link across different PS waveforms at different PS bands to provide voice and digital service. In this working mode, the CE will make a waveform solution according to the recognized environment information and user need, and deploy the solution through the radio platform. All the internal steps including radio environment profile fetching, solution lookup, solution profile generation and SDR waveform generation, are automated and not seen to the user. The user only needs to view existing link opportunities in the GUI display and pick one to talk. In the third mode, gateway mode, the PSCR will take two recognized but incompatible waveforms per use's choice, and bridge them together instead of talking to them individually as in talk mode.

To provide a platform independent user interface, a full Java GUI is developed for the PSCR node. Due to Java's virtual machine running mechanism [154], the GUI interface can be installed on various hardware platforms. As shown in Figure 7.4, the GUI also provides a network standard TCP socket interface to the CE to support distributed configuration and control.

In PSCR node the knowledge base is implemented as a standard MySQL database. It contains two types of relational tables as dictionaries for solution lookup. One is for the radio knowledge such as standard PS waveform specifications; and the other is the dictionary of parametric elements to be used to construct a solution by the solution maker. The knowledge base provides two management interfaces, one is a graphical web-based interface for local data management, such as external knowledge loading or update; the other is the standard open source Sequential Query Language (SQL) [155] interface over standard TCP socket to communicate with the solution maker in the CE. A detailed block diagram of PSCR knowledge base is shown in Figure 7.5. In the SQL database in Figure 7.5, the standard dictionary contains PS waveform knowledge such as band, channel tables, waveform formats, etc. the configuration dictionaries provide waveform and radio platform configuration components for the CE solution maker to look up and combine. The Apache server is used to provide web database interface.

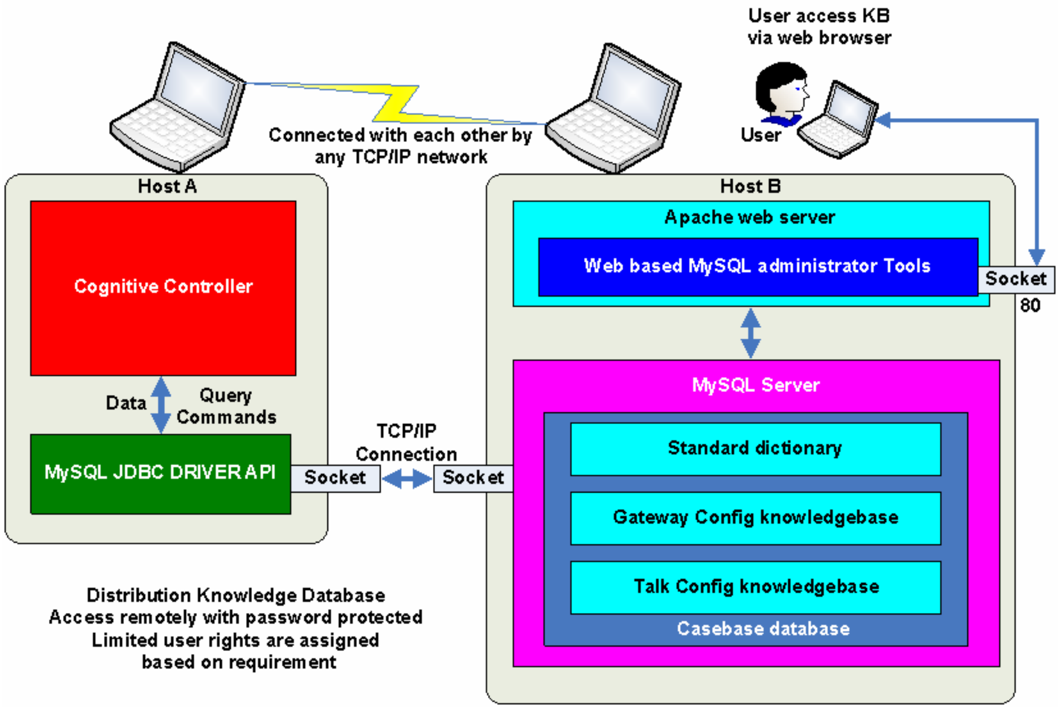


Figure 7.5: PSCR knowledge base subsystem block diagram

Look back to he CE subsystem in Figure 7.4, the CE is also implemented in Java to support platform independence. It also uses standard TCP sockets to communicate with GUI, radio platform, and knowledge database. Since it is written in Java, a Java database interface library, JDBC API, is used to connect CE with knowledge base via standard TCP socket.

To support use case 3, the PSCR is equipped with a platform independent radio interface, which means the PSCR CE can be easily ported on top of any radio platform given the radio's control interface written in a standard profile format. In current prototype system, CWT^2 waveform framework is configured to support required PS waveforms, which includes legacy FM, narrowband BPSK, QPSK, and MSK digital modulations at multiple PS bands. And for link control, carrier sense, push-to-talk, gateway, and broadcast MAC modules are provided to support different working modes. Figure 7.6 shows the detailed PS waveform framework. The flow graph can be configured to talk and gateway working mode with different PHY and MAC combinations. The scan mode still uses the standard spectrum probe and signal classifier modules as described in Chapter 6.

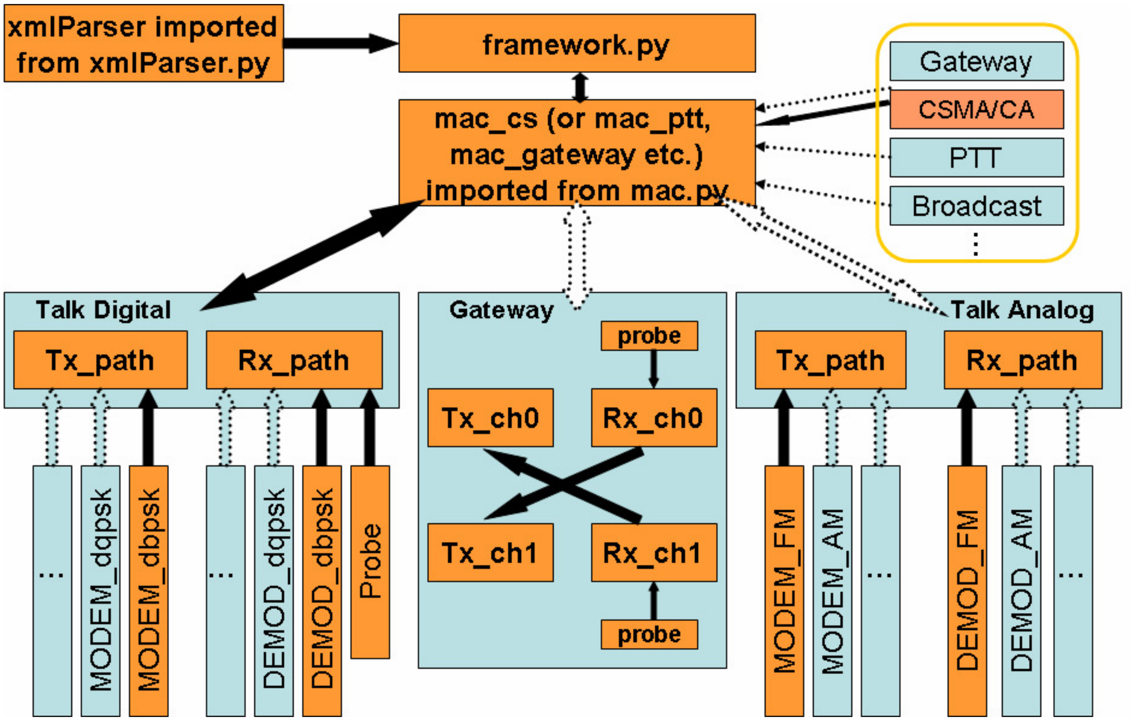


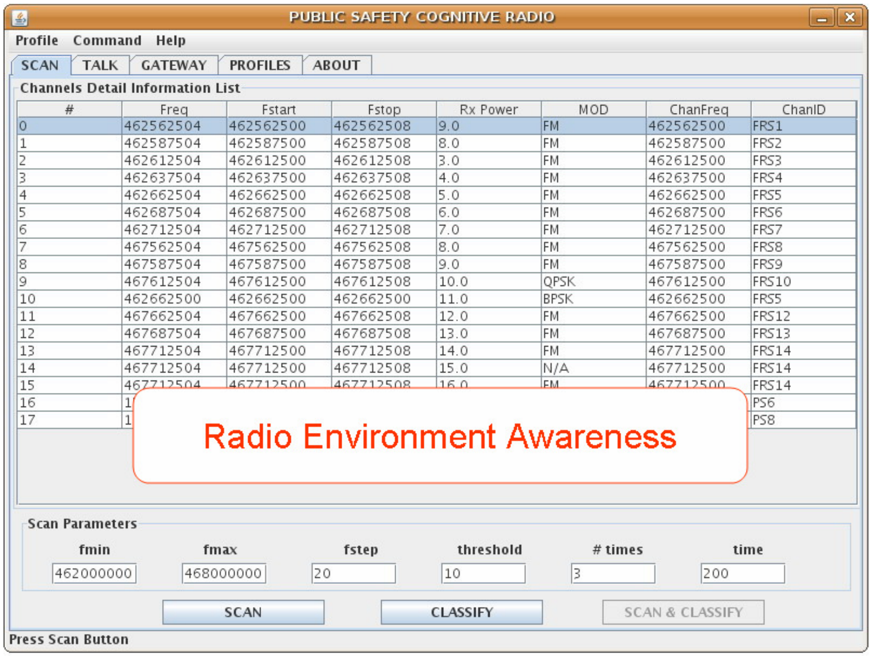
Figure 7.6: PSCR PHY/MAC waveform framework

The detailed PSCR node implementation, including GUI Java development, PS knowledge table definitions, SQL interface design, Java based sockets and system control threads, solution lookup, and PS specific CWT^2 waveform framework, are all included in the user manual in the PSCR node software package release v1.0, and they are indexed in appendix B.

7.3 PSCR Functionality Description with GUI Display

Following the previous section, a fully-functional PSCR node prototype was developed and demonstrated at the NIJ CommTech Program Review in Las Vegas on April 24, 2007. All three working modes, scan, talk, and gateway, were illustrated. A simple description of the PSCR node functionality with these three modes is given here with the designed GUI display.

Scan mode



Channels Detail Information List

#	Freq	Fstart	Fstop	Rx Power	MOD	ChanFreq	ChanID
0	462562504	462562500	462562508	9.0	FM	462562500	FRS1
1	462587504	462587500	462587508	8.0	FM	462587500	FRS2
2	462612504	462612500	462612508	3.0	FM	462612500	FRS3
3	462637504	462637500	462637508	4.0	FM	462637500	FRS4
4	462662504	462662500	462662508	5.0	FM	462662500	FRS5
5	462687504	462687500	462687508	6.0	FM	462687500	FRS6
6	462712504	462712500	462712508	7.0	FM	462712500	FRS7
7	467562504	467562500	467562508	8.0	FM	467562500	FRS8
8	467587504	467587500	467587508	9.0	FM	467587500	FRS9
9	467612504	467612500	467612508	10.0	QPSK	467612500	FRS10
10	462662500	462662500	462662500	11.0	BPSK	462662500	FRS5
11	467662504	467662500	467662508	12.0	FM	467662500	FRS12
12	467687504	467687500	467687508	13.0	FM	467687500	FRS13
13	467712504	467712500	467712508	14.0	FM	467712500	FRS14
14	467712504	467712500	467712508	15.0	N/A	467712500	FRS14
15	467712504	467712500	467712508	16.0	FM	467712500	FRS14
16	1						PS6
17	1						PS8

Scan Parameters

fmin	fmax	fstep	threshold	# times	time
462000000	468000000	20	10	3	200

Press Scan Button

Figure 7.7: PSCR node GUI in scan mode

On the scan mode GUI, the PSCR provides three options for the radio environment recognition. The first is a simple scan; the user specifies the frequency range (f_{min} and f_{max}) and frequency resolution (f_{step}). The energy threshold (in dB) for detection can be automatically adjusted by PSCR or manually set the user. The user can set the number of sweep times for an averaged result. The measurement time for each spectrum sweep can be set by the user as well. After all these are set, the PSCR controls the radio platform to perform the spectrum sweep, records detected spectrum components in a spectrum energy profile XML file, and sends it to GUI via TCP socket. The GUI then displays these energy components on the table. The second option is classify, which is based on the spectrum sweep result. A channel of interest is picked and the radio is instructed to collect waveform data from that channel for waveform recognition. Once the waveform is recognized, its format information including modulation and PS standard channel index are reported and displayed in GUI table. Option one and two can be combined into a batched operation, in which all the waveforms in the channels of interest are automatically classified one by one, and a summary of all the channel lists with related waveform formats are reported as a complete radio environment profile XML. The radio environment profile is need by the CE for solution making.

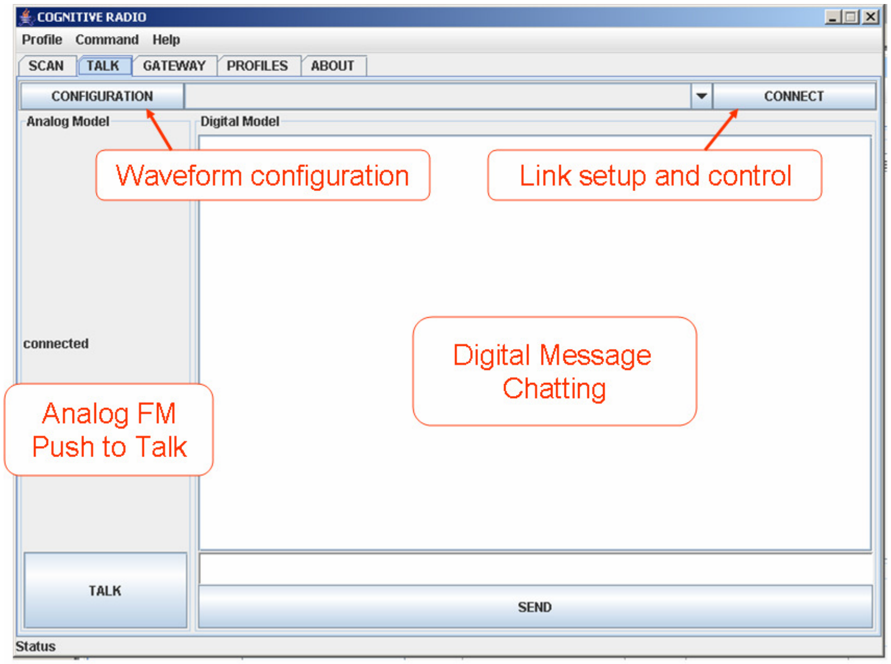
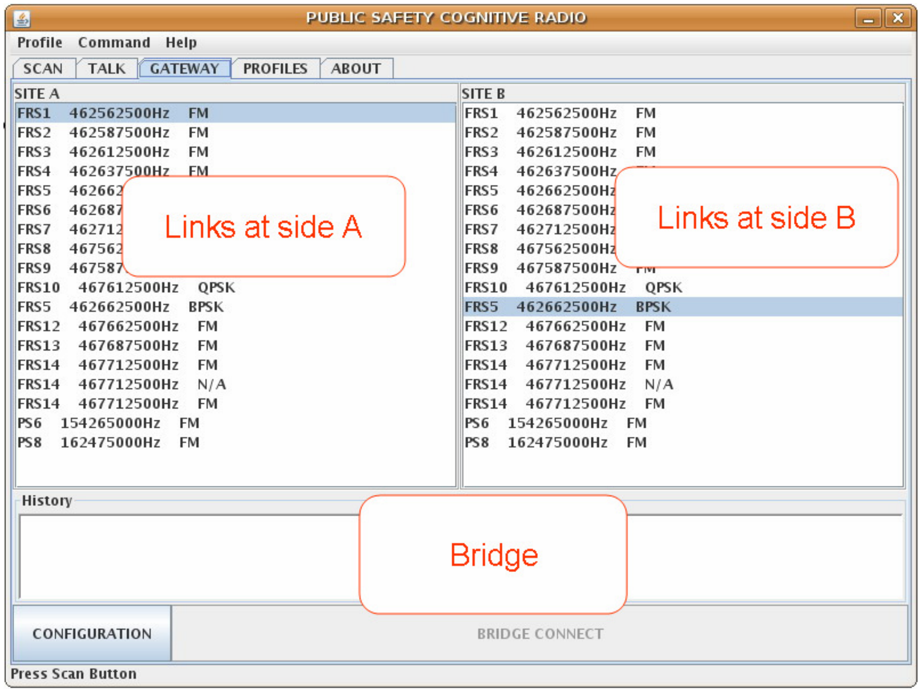


Figure 7.8: PSCR node GUI in talk mode

In talk mode, the PSCR GUI provides an interface for both voice and data services. According to the radio environment recognition result, the user can select one waveform desired to establish a link. For an analog waveform link, the voice link is established and voice push to talk interface is automatically activated for use. For a digital waveform link, a service offer by the GUI is a text messaging window. When a digital waveform link is selected by the user, the messaging window is activated and user can simply start chatting with the remote terminal. For both analog and digital services, the underlying waveform solution making, waveform PHY and MAC parameter definition, radio platform reconfiguration, waveform generation, and service activation, are all automatically managed by the PSCR CE.

Besides low rate digital text messaging, a real time video streaming with 350kbps data rate link is also illustrated between two PSCR nodes. The PSCR node can seamlessly sweep data rate from 20kbps to 500kbps with all supported digital modulations including BPSK, QPSK, and GMSK.



SITE A

```
FRS1   462562500Hz   FM
FRS2   462587500Hz   FM
FRS3   462612500Hz   FM
FRS4   462637500Hz   FM
FRS5   462662
FRS6   462687
FRS7   462712
FRS8   467562
FRS9   467587
FRS10   467612500Hz   QPSK
FRS5   462662500Hz   BPSK
FRS12   467662500Hz   FM
FRS13   467687500Hz   FM
FRS14   467712500Hz   FM
FRS14   467712500Hz   N/A
FRS14   467712500Hz   FM
PS6   154265000Hz   FM
PS8   162475000Hz   FM
```

SITE B

```
FRS1   462562500Hz   FM
FRS2   462587500Hz   FM
FRS3   462612500Hz   FM
FRS4   462637500Hz
FRS5   462662500Hz
FRS6   462687500Hz
FRS7   462712500Hz
FRS8   467562500Hz
FRS9   467587500Hz
FRS10   467612500Hz   QPSK
FRS5   462662500Hz   BPSK
FRS12   467662500Hz   FM
FRS13   467687500Hz   FM
FRS14   467712500Hz   FM
FRS14   467712500Hz   N/A
FRS14   467712500Hz   FM
PS6   154265000Hz   FM
PS8   162475000Hz   FM
```

Press Scan Button

Figure 7.9: PSCR node GUI in gateway mode

In gateway mode, given the radio environment information from scan mode, the user can simply pick two incompatible waveforms on the left and right lists. With a simple click of the configuration and bridge button, the PSCR node makes itself a waveform gateway bridging these two waveforms in real time. In the current PSCR node development, only the bridging between the same group of services are supported, such as between analog voice nodes or between digital nodes with similar throughput. One development work with gateway node on-going now is for bridging analog voice node with digital voice over IP node with the integration of voice codec module in the waveform framework platform. The voice codec is already developed and verified in normal digital links.

For more detailed information about the PSCR node demonstration, there is a video available for this PSCR node demonstration in the NIJ CommTech Program Review, which is available on the CWT web server. The complete PSCR node software package lease version 1.0 is also available on CWT web server with authorized download access.

7.4 Summary

In this chapter, a complete CR node system is integrated with all the building blocks designed from previous chapters. This complete CR node system serves as a general architecture solution that can be applied to various cognitive wireless applications. Based on this architecture, a PSCR node system is designed and implemented for public safety communication interoperability. It has complete recognition, solution making and learning flow. A standard MySQL database is implemented for the PSCR knowledge base. The complete PSCR node software system is packaged to an official release including installation guide and user/developer manuals. The functionality of this PSCR node is described through a demonstration.

Chapter 8: Conclusions and Recommendations

8.1 Summary of Research Results

This dissertation mainly includes the following five groups of contributions.

(1) **An architecture is defined for a cognitive radio node system.** This infrastructure features a platform independent software solution, which is important to bridge general machine learning algorithms with various radio hardware platforms. Inside the CR node infrastructure a cognition cycle is created specifically targeting the cognitive capabilities in radio domain. It has a two-loop hierarchical structure with outer layer of radio domain recognition and adaptation, and inner layer of machine reasoning and learning. Such a layered learning system is efficient in embedded system implementation and easy to be deployed with general radio and general cognitive applications. A "cognitive engine (CE)" term is borrowed from previous research and redefined as the cognitive software system in the CR architecture. The CE structure is fully defined to include the core set of cognitive algorithms, including environment recognition, solution making, and self-learning, to serve as the software solution working with reconfigurable radio platforms to form general cognitive radio node systems. The CE structure is also designed to have an open, modular structure to support both node and network based cognitive functionalities. With this CR architecture, a public safety application specific cognitive radio node is implemented and prototyped.

(2) **Following the CR architecture, the key functional modules of the CE are designed and implemented**. The first module is the radio environment recognition, which is important for the CR to achieve the environment awareness. An adaptive block FFT spectrum sweeper is designed for energy detection across the frequency

range of interest, and a two-stage adaptive signal classification signal is designed to identify the waveform in the channel of interest. Also a modulation adaptive synchronization system is designed including carrier recovery, phase lock, and general symbol timing. They are implemented based on GNU Radio software package. The second module designed is the learning core, which is an integration of case-based reasoning (CBR) and genetic algorithm (GA). The combination of CBR and GA chain enables a balance between rational, general solution look-up, which is suitable for known problem scenarios, and creative solution search for novel situations. Both the solution making methods converge on the reinforced learning approach by updating the knowledge with the feedback from the practice. Knowledge from CR related domains are defined as knowledge profiles and described in XML to serve the intelligent information processing. And a CR knowledge based is designed and implemented as a standard SQL database to support knowledge storage and data mining for the learning core.

(3) **To make the designed CR system a platform independent system solution, a general radio interface is designed to enable hardware-independent radio resource configuration and control to serve the cognitive algorithms for various cognitive wireless applications.** Such a radio interface is structured with configuration thread and control thread. They are independent from each other to achieve the maximal flexibility and reconfigurability. Accordingly, both configuration and control are encoded in waveform and radio platform profiles described in XML transferring between the CE and the radio. The information transfer is implemented by standard TCP socket method so network based configuration and control is also available.

(4) **To support and verify CE, a highly-flexible SDR platform, called "waveform framework", is designed and implemented using GNU Radio software package.** The waveform framework follows modular design to support reconfigurable PHY and MAC runtime reconfiguration and adaptation. For the radio resource management,

both waveform configuration and link control are designed with multi-thread hierarchy for maximal flexibility and processing efficiency.

(5) **A fully functional Public Safety Cognitive Radio (PSCR) node prototype is designed and implemented for the public safety interoperability.** It consists of scan, talk and gateway modes for different service types and provides all the required PS waveforms. In this prototype, the cognitive engine solution making core and the user interface are implemented in Java, thus supports platform independent operation. With standard interface TCP sockets also implemented in Java, the PSCR node is suitable for network based distributed cognitive functionalities.

A list of publications related to this dissertation is given in Table 8.1.

Table 8.1: Related research publications

1. B. Le, Building a Cognitive Radio: From Architecture Definition to Prototype Implementation, Ph.D. dissertation, Virginia Tech. 2007
2. B. Le, P. Garcia, Q. Chen, B. Li, F. Ge, M. Elnainay, T. W. Rondeau and C. W. Bostian, A Public Safety Cognitive Radio Node System, to appear at SDR Forum conference, Denver, CO, Nov. 2007
3. B. Le, T. W. Rondeau and C. W. Bostian, General Radio Interface Between Cognitive Algorithms and SDR Platforms, to appear at SDR Forum conference, Denver, CO, Nov. 2007
4. B. Le, Adaptive Waveform Recognition For Cognitive Radio Nodes, Virginia Tech Intellectual Property (VTIP), under application, 2007
5. B. Le, Platform Independent Radio Interface for Cognitive Radio Nodes, Virginia Tech Intellectual Property (VTIP), under application, 2007
6. K. E. Nolan, P. D. Sutton, L. E. Doyle, T. W. Rondeau, B. Le, C. W. Bostian, Dynamic Spectrum Access and Coexistence Experience Involving Two Independently Developed Cognitive Radio Testbeds, Proc. DySPAN 2007.
7. K. E. Nolan, P. D. Sutton, L. E. Doyle, T. W. Rondeau, B. Le, C. W. Bostian, Demonstration and Analysis of Collaboration, Coexistence, and

Interoperability of Cognitive Radio Platforms, IEEE Consumer Communications and Networking Conference (CCNC) invited paper, 2007.

8. T. W. Rondeau, K. E. Nolan, D. H. Friend, B. Le, A. B. MacKenzie and C. W. Bostian, Architecture for a Cognitive Engine: Including Extensibility and Distributed Processing, IEEE Global Communications (GLOBECOM) conference, Washington DC, Nov. 07.

9. B. Le, T. W. Rondeau and C. W. Bostian, Cognitive Radio Reality, Journal on Cognitive Radio special issue of Wireless Communications and Mobile Computing, to appear fall 2007.

10. Y. Zhao, B. Le and J. H. Reed, Network Support – the Radio Environment Map, Chapter 11, Cognitive Radio Technology, Ed. Bruce Fette, Newnes, 2006.

11. B. Le, T. W. Rondeau, D. Maldonado, D. Scaperoth and C. W. Bostian, Signal Recognition for Cognitive Radios, Proc. Software Defined Radio Forum (SDRF) Conference, Orlando, FL, Nov. 06.

12. T. W. Rondeau, B. Le, D. Maldonado, D. Scaperoth, A. B. MacKenzie and C. W. Bostian, Optimization, Learning and Decision Making in Cognitive Systems, Proc. SDR Forum Conference, Orlando, FL, Nov. 06.

13. C. W. Bostian and B. Le, Cognitive Radio and Public Safety Communications, APCO 2006 International Conference, Orlando, FL, Aug. 2006

14. D. Scaperoth, B. Le, T. W. Rondeau, D. Maldonado and C. W. Bostian, Cognitive Radio Platform Development for Interoperability, Proc. Military Communications Conference (MILCOM), Washington DC, Oct. 2006

15. B. Le, Recognition and Adaptation in PHY Layer, SDR Forum cognitive radio workshop, San Francisco, CA, Apr. 2006

16. T. W. Rondeau, B. Le, D. Maldonado, D. Scaperoth and C. W. Bostian, Cognitive Radio Formulation and Implementation, Proc. IEEE CrownCom. Conference, Mykonos Island, Greece, Jun. 2006.

17. B. Le, T. W. Rondeau, J. H. Reed and C. W. Bostian, Analog-to-Digital Converters, Past, Present and Future. Top 3 downloaded paper on IEEE Signal Processing Magazine, Nov. 2005.

18. D. Maldonado, B. Le, A. Hugine, T. W. Rondeau and C. W. Bostian, Cognitive Radio Applications: Vision and Example, IEEE Dynamic Spectrum Access Network (DySPAN) conference, Baltimore, NJ, Nov. 2005.

19. B. Le, T. W. Rondeau, D. Maldonado and C. W. Bostian, Modulation Identification Using Neural Networks for Cognitive Radios, Proc. SDR Forum Conference, Anaheim, CA, Nov. 2005.

20. B. Le, T. W. Rondeau, D. Maldonado and C. W. Bostian, Radio-Domain Cognition – a System Approach in Theory and Implementation, Virginia Tech Intellectual Property (VTIP) No. 05-077, Oct. 2005.

21. T. W. Rondeau, C. W. Bostian, D. Maldonado, A. Ferguson, S. Ball, B. Le and S. Midkiff, Cognitive Radios in Public Safety and Spectrum Management, Proc. Conf. Communication, Information and Internet Policy, Sep. 2005

22. T. W. Rondeau, B. Le, C. J. Rieser and C. W. Bostian, Cognitive Radios with Genetic Algorithms: Intelligent Control of Software Defined Radios. Outstanding paper award, Proc. SDRF Conference, Nov. 2004

Table 8.2 provides a list of CR research and development system demonstrations.

Table 8.2: Related system development demonstrations

1. Signal classification system NIJ project meeting demonstration, 2005. (See Section 3.9.1)
2. Smart receiver and waveform framework Anritsu demonstration at SDRF conference 2006. (See Section 3.9.2)
3. PSCR node prototype NIJ CommTech conference demonstration 2007. (See Chapter 7).

8.2 Future Research Recommendations

This dissertation serves to help build the research and engineering foundations of the CWT^2 cognitive radio system development. With the current development of CWT^2 CR node system, there are several opportunities for future academic exploration.

Based on current PHY and MAC capability of the CWT^2 CR node, a general cognitive radio MAC frame format is needed to help standardize cognitive link adaptation procedure. And a corresponding standard adaptation protocol is needed for cognitive network functionalities. On top of the currently developed MAC layer, a Logic Link Control (LLC) is to be designed to enhance the cross-layer cognitive performance optimization.

In the PSCR node, even for the standard waveform solutions, the GA part of the solution making may need to be activated to improve the link performance. To realize this, the related performance parameters should be identified and encoded in GA fitness functions to guide the solution search. And the solution space boundary should be carefully set to be within the scope of standard waveform specifications. A set of metrics will also need to be defined to benchmark the performance gain with such optimization.

In the current CWT^2 waveform framework platform, the waveform configuration and link control are carried by Python multi-threading, which is relatively resource intensive. Porting these Python threads to C++ will greatly improve system real-time response with much less computation and memory cost. Also, it would be an interesting practice to port the waveform framework – even the whole CR node software system - to a multi-core processing platform such as the Cell Broadband Engine to claim more power for better real-time response.

Bibliography

[1] G. Dimitrakopoulos, P. Demestichas, D. Grandblaise, K. Mobner, J. Hoffmeyer, and J. Luo, "Cognitive radio, spectrum and radio resource management," *WWRF White Paper WG6-wp4-v2004,* 2004.

[2] B. Fette, *Cognitive Radio Technology*. New York: Newnes, 2006.

[3] S. Haykin, "Cognitive Radio: Brain-Empowered Wireless Communications," *IEEE Journal on Selected Areas in Communications,* vol. 23, pp. 201 - 220, Feb. 2005.

[4] J. Mitola, "Cognitive Radio for Flexible Mobile Multimedia Communications," in *IEEE International Workshop on Mobile Multimedia Communications*, 1999, pp. 3 – 10.

[5] C. W. Bostian, S. F. Midkiff, T. M. Gallagher, C. J. Rieser, and T. W. Rondeau, "Rapidly Deployable Broadband Communications for Disaster Response," in *Sixth International Symposium on Advanced Radio Technologies (SAFECOM Session)*, Boulder, CO, 2004 pp. 87-92.

[6] S. F. Midkiff and C. W. Bostian, "Rapidly deployable broadband wireless communications for emergency managment," *National Digtal Government Research Conference,* May 21-23 2001.

[7] C. J. Rieser, "Biologically Inspired Cognitive Radio Engine Model Utilizing Distributed Genetic Algorithms for Secure and Robust Wireless Communications and Networking," *Ph. D. Dissertation, Electrical and Computer Engineering Dept. Virginia Tech,* 2004.

[8] J. Mitola, "Cognitive Radio: An Integrated Agent Architecture for Software Defined Radio," Royal Institute of Technology (KTH), 2000.

[9] C. J. Rieser, T. Rondeau, C. Bostian, W. Cyre, and T. Gallagher, "Biologically Inspired Cognitive Wireless L12 Technology: Genetic Algorithms Applied to Cognitive Radio," *Virginia Tech IP (VTIP) No. 03-056,* March 16 2004.

[10] DARPA: DARPA-XG RFCS: http://www.darpa.mil/ato/programs/xg/rfcs.htm.

[11] FCC: Spectrum Policy Task Force (SPTF): http://www.fcc.gov/sptf/.

[12] FCC, "In the Matter of Facilitating Opportunities for Flexible, Efficient, and Reliable Spectrum Use Employing Cognitive Radio Technologies, Authorization and Use of Software Defined Radios," *FCC Notice for Proposed Rule Making (NPRM) 03-322,* Dce. 30 2003.

[13] FCC, "Facilitating Opportunities for Flexible, Efficient, and Reliable Spectrum Use Employing Cognitve Radio Techniques," *FCC Notice and Order (NAO) 05-57,* March 11 2005.

[14] IEEE: 802.22 standards group web site: http://www.ieee802.org/22/ 2004.

[15] IEEE: P1900.X web site: http://www.ieeep1900.org/, 2005.

[16] FCC, "Office of engineering and technology announces project schedule for proceeding on unlicensed operation in the TV broadcast bands," *Public Notice ET Docket No. 04-186,* Sep. 11 2006.

[17] NSF: NetS web: http://www.nsf.gov/pubs/2005/nsf05505/nsf05505.htm.

[18] NIJ: CommTech web: http://www.ojp.usdoj.gov/nij/topics/technology/communication/welcome.htm.

[19] C. J. Rieser, T. W. Rondeau, and C. W. Bostian, "Cognitive Radio Testbed: Further Details and Testing of a Distributed Genetic Algorithm Based Cognitive Engine for Programmable Radios," in *IEEE MILCOM*, Monterey, CA., 2004, pp. 1437 - 1443

[20] B. Le and C. W. Bostian, "Radio Domain Cognition - a System Approach in Theory and Implementation " *Virginia Tech IP (VTIP) No. 05-077,* Oct. 31 2005.

[21] B. Le, T. W. Rondeau, D. Maldonado, D. Scaperoth, and C. W. Bostian, "Siganl recognition for cognitive radios," in *Software Defined Radio Forum Conf.* Orlando, FL, 2006.

[22] R. W. Thomas, L. A. Dasilva, and A. B. MacKenzie, "Cognitive networks," in *IEEE DySPAN*, Baltimore, MD, 2005, pp. 352-360.

[23] K. E. Nolan, T. W. Rondeau, P. Sutton, B. Le, L. Doyle, and C. W. Bostian, "Demonstration and analyses of collaboration, co-existence and interoperability of cognitive radio platforms," in *IEEE 1st workshop on Cognitive Radio Networks (CCNC)*, Las Vegas, 2007.

[24] P. Sutton, L. Doyle, and K. E. Nolan, "A reconfigurable platform for cognitive networks," in *IEEE CROWNCOM*, Mykonos, Greece, 2006.

[25] K. E. Nolan, T. W. Rondeau, P. Sutton, C. W. Bostian, and L. Doyle, "A framework for implementing cognitive functionality," in *SDRF conf.*, Orlando, FL, 2006.

[26] B. A. Fette, *Cognitive Radio Technology*: Newnes, 2006.

[27] CWT, "A public safety cognitive radio for universal interoperability," Center for Wireless Telecommunications, Wireless at Virginia Tech, NIJ CommTech program proposal 2006.

[28] T. A. Sturman, "An evaluation of software defined radio - main document," in *OFCOM R&D Symposium*, London, 2006.

[29] B. Le, T. W. Rondeau, D. Maldonado, and C. W. Bostian, "Modulation Identification Using Neural Network for Cognitive Radios," in *Software Defined Radio Forum Technical Conference*, Anaheim, CA, 2005.

[30] S. Russell and E. Wefald, "Principles of metareasoning," *Artificial Intelligence,* vol. 49, pp. 361-395, 1991.

[31] T. W. Rondeau, B. Le, C. J. Rieser, and C. W. Bostian, "Cognitive Radios with Genetic Algorithms: Intelligent Control of Software Defined Radios," in *Software Defined Radio Forum Technical Conference*, Phoenix, AZ. , 2004, pp. C-3 - C-8.

[32] D. Maldonado, B. Le, A. Hugine, T. W. Rondeau, and C. W. Bostian, "Cognitive radio application to dynamic spectrum allocation: a discussion and an illustrative example," in *IEEE DySPAN*, Baltimore, MD, 2005.

[33] Y. Zhao, L. Morales, J. Gaeddert, K. K. Bae, J-S. Um, and J. H. Reed, "Applying radio environment maps to cognitive wireless regional area networks," in *IEEE DySPAN*, Dublin, Irland, 2007.

[34] GNU-Radio: GNU software project web: http://www.gnu.org/software/gnuradio/.

[35] B. Le, T. W. Rondeau, J. H. Reed, and C. W. Bostian, "Analog-to-digital converters: a review of the past, present and future," *IEEE Signal Processing Magazine,* vol. 22, pp. 69-77, Nov. 2005.

[36] C. W. Bostian and J. H. Reed, "Trade study of implementation of SDR: fundamental limitations and future prospects," Wireless at Virginia Tech, ECE dept. Virginia Tech 2006.

[37] A. F. Naguib, V. Tarokh, N. Seshadri, and A. R. Calderbank, " Space-time coded modulation for high data rate wireless communications," in *IEEE GLOBECOM*, 1997.

[38] K. D. Jong, "On decentralizing selection algorithms," in *6th International Conf. Genetic Algorithms (ICGA)*, Morgan Kaufmann, San Francisco, CA, 1995.

[39] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*: MIT Press, 1998.

[40] Y. Zhao, J. Gaeddert, K. K. Bae, and J. H. Reed, "Radio environment map enabled situation-aware cognitive radio learning algorithms," in *Software Defined Radio Forum (SDRF) technical conference*, Orlando, FL, 2006.

[41] SCA: JTRS SCA standard web: http://jtrs.spawar.navy.mil/sca/.

[42] L. W. Couch, *Digital and Analog Communication Systems*, 6 ed.: Prentice Hall, 2001.

[43] S. M. Kay and S. L. Marple, "Spectrum analysis-A modern perspective," *Proceedings of the IEEE,* vol. 69, pp. 1380 - 1419, 1981

[44] W. A. Gardner, "Signal Interception: A Unifying Theoretical Framework for Feature Detection," *IEEE Trans. Communications,* vol. 36, pp. 897 - 906, Aug. 1988.

[45] C. M. Spooner, W. A. Brown, and G. K. Yeung, "Automatic radio-frequency environment analysis," in *34th IEEE conf. ASILOMAR*, Pacific Grove, CA, 2000.

[46] A. Fehske, J. Gaeddert, and J. H. Reed, "A new approach to signal classification using spectrul correlation and neural networks," in *IEEE DySPAN*, Baltimore, MD, 2005.

[47] C. M. Spooner, "Classification of co-channel communication signals using cyclic cumulants," in *IEEE conf. ASILOMAR*, 1996.

[48] D. Cabric, S. M. Mishra, and R. W. Brodersen, "Implementation Issues in Spectrum Sensing
for Cognitive Radios," in *IEEE Proc. Signals, Systems and Computers*, 2004, pp. 772 - 776

[49] S. L. Marple, "A tutorial overview of modern spectral estimation," in *IEEE Proc. ICASSP*, 1989, pp. 2152 - 2157.

[50] N. Ghani and R. Lamontagne, "Neural networks applied to the classification of spectral features for automatic modulation recognition," in *IEEE Proc. MILCOM*, 1993 pp. 111 - 115.

[51] J. G. Proakis, *Digital Communications*, 4 ed. New York: McGraw Hill, 2000.

[52] R. Bracewell, *The Fourier Transform and Its Applications*, 2nd ed.: McGraw-Hill, 1986.

[53] B. F. Beidas, "Higher order correlation-based approach to modulation classification of digital frequency modulated signals," *IEEE Journal on Selected Areas In Communications,* vol. 13, Jan 1995.

[54] C.-Y. Huang, "Likelihood methods for MPSK modulation classification," *IEEE Trans. Communications,* vol. 43, Feb/Mar/Apr 1995.

[55] B. F. Beidas, "Asynchronous Classification of MFSK signals using higher-order correlation domain," *IEEE Trans. Communications,* vol. 46, Apr. 1998.

[56] A. Polydoros, "On the detection and classification of quadrature digital modulations in broad-band noise," *IEEE Trans. Communications,* vol. 38, Aug. 1990.

[57] S.-Z. Hsue and S. S. Soliman, "Automatic modulation classification using zero crossing," *IEEE Proc. Radar and Signal Processing,* vol. 137, pp. 459-464, Dec. 1990 1990.

[58] I. Druckmann, E. I. Plotkin, and M. N. S. Swamy, "Automatic modulation type recognition," in *IEEE Canadian Conf. Electrical and Computer Engineering*, 1998, pp. 65-68.

[59] F. F. Liedtke, "Computer simulation of an automatic classification procedure for digitally modulated communication signals with unknown parameters," *Signal Processing,* vol. 6, pp. 311-323, Aug 1984.

[60] F. Jondral, "Automatic classification of high frequency signals," *Signal Processing,* vol. 9, pp. 177-190, Oct. 1985.

[61] R. J. Mammone, R. J. Rothaker, C. I. Podilchuk, S. Davidovici, and D. L. Schilling, "Estimation of carrier frequency, modulation type and bit rate of an unknown modulated signal," in *International Communications Conference (ICC)*, 1987.

[62] L. V. Dominguez, J. M. P. Borrallo, and J. P. Garcia, "A general approach to the automatic classification of radiocommunication signals," *Signal Processing,* vol. 22, pp. 239-250, Mar. 1991.

[63] P. A. J. Nagy, "Modulation classification - a unified view," in *EUSIPCO'96*, 1996.

[64] C. Schreyogg and J. Reichert, "Robust classification of modulation types using spectral features applied to HMM," in *IEEE MILCOM*, 1997, pp. 1377-1381.

[65] J. E. Whelchel, D. L. McNeill, R. D. Hughes, and M. M. Loos, "Signal understanding: an artificial intelligence approach to modulation classification," in *IEEE Int. Workshop on Tools for AI, Architecture, Language and Algorithms*, 1989, pp. 231-236.

[66] A. K. Nandi and E. E. Azzouz, "Algorithms for Automatic Modulation Recognition of Communication Signals," *IEEE Trans. Communications,* vol. 46, pp. 431 - 436, Apr. 1998.

[67] E. E. Azzouz and A. K. Nandi, "Procedure for automatic recognition of analogue and digital modulations," *IEE Proc. Communications,* vol. 143, pp. 259 - 266, Oct. 1996

[68] Z. Yaqin, R. Guanghui, W. Xuexia, W. Zhilu, and G. Xuemai, "Automatic digital modulation recognition using artificial neural networks," in *IEEE Proc. Neural Networks and Signal Processing*, 2003 pp. 257 - 260.

[69] G. Arulampalam, V. Ramakonar, A. Bouzerdoum, and D. Habibi, "Classification of Digital Modulation Schemes Using Nerual Networks," in *International Symposium Signal Processing and its Applications*, Brisbane, Australia, 1999, pp. 649 - 652.

[70] C. Louis and P. Sehier, "Automatic modulation recognition with a hierarchical neural network," in *IEEE Proc. MILCOM*, 1994 pp. 713 - 717.

[71] N. Kim, N. Kehtarnavaz, M. B. Yeary, and S. Thornton, "DSP-based hierarchical neural network modulation signal classification," *IEEE Trans. Neural Networks,* vol. 14, pp. 1065 - 1071, Sept. 2003

[72] W. A. Gardner, *Statistical Spectral Analysis - A Nonprobabilistic Theory*. Englewood Cliffs, NJ: Prentice-Hall, 1988.

[73] W. A. Gardner, W. A. Brown, and C. Chen, "Spectral correlation of modulated signals: part II - digital modulation," *IEEE Trans. Communications,* vol. COM-35, pp. 595-601, June 1987.

[74] W. A. Gardner, "Spectral correlation of modulated signals: part I - analog modulation," *IEEE Trans. Communications,* vol. COM-35, pp. 584-594, 1987.

[75] K. Kim, I. A. Akbar, K. K. Bae, J.-S. Um, C. M. Spooner, and J. H. Reed, "Cyclostationary approaches to signal detection and classification in cognitive radio," in *IEEE DySPAN*, Dublin, Irland, 2007.

[76] L. E. Franks, "Carrier and Bit Synchronization in Data Communication - A Tutorial Review," *IEEE Trans. Communications,* vol. COM-28, No. 8, pp. 1107-1120, 1980.

[77] S. U. H. Qureshi, "Adaptive Equalization," *Proceedings of the IEEE,* vol. 73, September 1985 1985.

[78] J. R. Treichler, I. Fijalkow, and J. C. R. Johnson, "Fractionally Spaced Equalizers: How Long Should They Really Be? ," in *IEEE Signal Processing Magazine*, 1996.

[79] L. R. Litwin, "Adaptive channel equalization for wireless communications," *Ieee Potentials,* vol. 18, pp. 13-15, Apr-May 1999.

[80] L. R. Litwin, "Blind channel equalization," *IEEE Potentials,* vol. 18, pp. 9-12, Oct-Nov 1999.

[81] R. Gooch and J. Lundell, "The CM Array: An Adaptive Beamformer for Constant Modulus Signals," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 1986, pp. 2523 - 2526.

[82] B. G. Agee, "The Property-Restoral Approach to Blind Adaptive Signal Extraction," in *Electrical Engineering and Computer Science* vol. Doctoral: Univeristy of California, Davis, 1989.

[83] B. G. Agee, "Blind Separation and Capture of Communication Signals Using a Multitarget Constant Modulus Beamformer," *Military Communications Conference (MILCOM),* vol. 2, pp. 340-346, 15-18 Oct. 1989.

[84] B. G. Agee, "The Least-Square CMA: A New Technique for Radio Correction of Constant Modulus Signals " in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP).* Tokyo, 1986, pp. 953-956.

[85] J. R. Treichler and B. G. Agee, "A New Approach to Multipath Correction of Constant Modulus Signals," *IEEE Trans. on Acoustics, Speech, and Signal processing,* vol. ASSP-31, pp. 459-472, April 1983.

[86] J. R. Treichler and M. G. Larimore, "New Processing Techniques Based on the Constant Modulus Adaptive Algorithms," *IEEE Trans. Acoustic, Speech, and Signal Processing,* vol. ASSP-33, pp. 420-431, April 1985.

[87] S. Barbarosa and A. Scaglione, "Blind equalization using cost function matched to the signal constellation," in *IEEE 31th ASILOMAR conf. Signals, Systems, & Computers*, 1997, pp. 550-554.

[88] D. N. Godard, "Slef-Recovering Equalization and Carrier Tracking in Two-Dimensional Data Commonnications Systems," *IEEE Trans. Communications,* vol. Com-28, pp. 1867-1875, November 1980.

[89] L. Tong, G. Xue, and T. Kailath, "Blind identification and equalization based on second-order statistics: a time domain approach," *IEEE Trans. Information Theory,* vol. 40, pp. 340-349, Mar. 1994.

[90] A. Sahai, N. Hoven, and R. Tandra, "Some fundamental limitations on cognitive radio," in *Allerton Conf.* , Monticello, 2004.

[91] A. V. Rosti and V. Koivunen, "Classification of MFSK modulated signals using the mean of complex envelope," in *Proc. EUSIPCO-2000*, 2000, pp. 581-584.

[92] J. Lopatka and M. Pedzisz, "Automatic modulation classification using statistical moments and a fuzzy classifier," in *Signal Processing Proceedings, WCCC-ICSP*, 2000, pp. 1500-1506.

[93] P. Welch, "The use of fast Fourier transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms," *IEEE Trans. Audio and Electroacoustics,* vol. 15, pp. 70-73, June 1967.

[94] f. harris, "DSP based carrier and timing synchronization in digital modems," in *Software Defined Radio Conf.* , Anahaim, CA, 2005.

[95] A. Leclert, "Universal carrier recovery loop for QASK and PSK signal sets," *IEEE Trans. Communications,* vol. COM-31, pp. 130-136, Jan 1983.

[96] M. Filip and K. H. Pang, "Simulation of a universal base-band carrier recovery loop," in *IEE Colloquium on New Synchronisation Techniques for Radio Systems*, 1995, pp. 5/1 - 5/6.

[97] f. harris, "DSP based carrier and timing synchronization in digital modems," in *Proc. SDR Forum 05 Technical Conference*, Anahaim, CA, 2005.

[98] Y. Miyake, M. Hagiwara, and M. Nakagawa, "A new timing extraction method and data interpolation for block demodulation," in *IEEE ICASSP*, 1989, pp. 1223-1226.

[99] S. Taira and E. Murakami, "Automatic classification of analogue modulation signals by statistical parameters," in *IEEE MILCOM*, 1999, pp. 202-207.

[100] A. K. Jain, R. P. W. Duin, and J. Mao, "Statistical pattern recognition: a review," *IEEE TRans. Pattern Analysis and Machine Intelligence,* vol. 22, pp. 4-37, 2000.

[101] S. Y. Kung, *Digital Neural Networks*, 1st ed.: Prentice Hall, 1997.

[102] B. G. Mobasseri, "Constellation Shape as a Robust Signature for Digital Modulation Recognition," in *Military Communications*, Atlantic City, NJ, 1999, pp. 442 - 446.

[103] K. Umebayashi and R. H. Morelos-Zaragoza, "A method of non-data-aided carrier recovery with modulation identification," in *Proc. IEEE Global Telecommunication Conference*, San Antonio, Texas, 2001, pp. 3375-3379.

[104] M. Filip and K. H. Pang, "Simulation of a universal base-band carrier recovery loop," in *IEE Colloquium on New Synchronisation Techniques for Radio Systems*, 1995.

[105] H. Meyr, M. Moeneclaey, and S. A. Fechtel, *Digital Communication Receivers - Synchronization, Channel Estimation, And Signal Processing*: John Wiley & Sons, 1998.

[106] K. H. Mueller and M. S. Muller, "Timing recovery in digital synchronous data receivers," *IEEE Trans. Communications,* vol. COM-24, pp. 516-531, May 1976.

[107] J. Hawkins, *On Intelligence*. New York: Times Books, 2004.

[108] M. Negnavitsky, *Artificial Intelligence: A Guide to Intelligent Systems*. Harlow, England: Addison-Wesley, 2002.

[109] I. Gilboa and D. Schmeidler, *A Theory of Case-Based Decisions*. Cambridge: Cambridge University Press, 2001.

[110] J. Kolodner, *Case-Based Reasoning*. San Mateo, CA: Morgan Kaufmann Pub., 1993.

[111] F. Streichert, *Evolutionary Algorithms in Multi-Modal and Multi-Objective Environments*: Logos Verlag Berlin, 2007.

[112] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA: Addison-Wesley, 1989.

[113] T. Back, *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*: Oxford Univ. Press, 1996.

[114] W. M. Spears, K. A. D. Jong, T. Back, D. B. Fogel, and H. d. Garis, "An overview of evolutionary computation," in *European Conf. Machine Learning*, 1993.

[115] M. Srinivas and L. M. Patnaik, "Genetic algorithms: a survey," *IEEE Computer,* vol. 27, pp. 17-26, June 1994.

[116] B. Le, T. W. Rondeau, and C. W. Bostian, "Cognitive radio realities," *Cognitive Radio special issue, Journal on Wireless Communications and Mobile Computing* to appear fall 2007.

[117] R. L. d. Mantaras and E. Plaza, "Case-based reasoning: an overview," *AI Communications,* 1997.

[118] M. Mitchell, *An Introduction to Genetic Algorithms*. Cambridge, MA: MIT Press, 1996.

[119] C. M. Fonseca and P. J. Fleming, "Genetic Algorithms for multiobjective optimization: formulation, discussion, and generalization," in *Proc. Int. Conf. Genetic Algorithms*, 1993, pp. 416 – 423.

[120] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms - a comparative case study and the strength Pareto approach," *IEEE Trans. Evolutionary Computation,* vol. 3, pp. 257 - 271, 1999.

[121] T. W. Rondeau, B. Le, D. maldonado, D. Scaperoth, and C. W. Bostian, "Cognitive radio formulation and implementation," in *IEEE Proc. CROWNCOM*, Mykonos, Greece, 2006, pp. pp 1-10.

[122] FCC, "Part 15 - radio frequency devices," *FCC Rules and Rulations, Part 15,* pp. 1-155, 2006.

[123] Y. Chen, "Bayesian hierarchical modelling of dual response surfaces," *Ph. D. Dissertation, Statistics Dept. Virginia Tech,* Nov. 2005.

[124] E. Hohveldin, "Communication profiles," in *Software Defined Radio: Architectures, Systems and Functions*, M. Dilinger, Ed.: Hohn Wiley & Son Ltd, 2003.

[125] J. Wiles and J. Hallinan, *Computational Intelligence*: Whiley-Interscience, 2003.

[126] W3C: World Wide Web Consortium XML section: http://www.w3.org/XML/

[127] OMG: UML homepage: http://www.omg.org/gettingstarted/what_is_uml.htm.

[128] SDRF, "Hardware abstraction layer workgroup report on results of request for information," *SDRF RFI doc: SDRF-04-A-0009-V0.00,* July 17 2004.

[129] D. Scaperoth, B. Le, T. W. Rondeau, D. Maldonado, and C. W. Bostian, "Cognitive radio platform development for interoperability," in *Military Communications Conference (MILCOM)* Washington D. C., 2006.

[130] ITU-R, "WP8A: Draft Report on SDR," Sep. 2004.

[131] J. H. Reed, *Software Radio: A Modern Approach to Radio Engineering,* Englewood Cliffs, NJ: Prentice-Hall, 2004.

[132] F. K. Jondral, "Software-defined radio - basics and evolution to cognitive radio," *EURASIP Journal on Wireless Communications and Networking,* pp. 275-283, Mar. 2005.

[133] IMEC, "Software defined radios tutorial part II: enabling flexibility for low power " in *IEEE CROWNCOM*, Mykonos, Greece, 2006.

[134] R. H. Walden, "Analog-to-digital converter survey and analysis," *IEEE Journal on Selected Areas In Communications,* vol. 17, pp. 539-550, Apr. 1999.

[135] D. B. Chester, "Digital IF filter technology for 3G systems: an introduction," *IEEE Signal Processing Magazine,* Feb. 1999.

[136] A. Gatherer, T. Stetzler, M. McMahan, and E. Auslander, "DSP-based architectures for mobile communications: past, present and future," *IEEE Communications Magazine,* vol. 38, pp. 84-90, Jan. 2000.

[137] IBM: Cell Broadband Engine homepage: http://www.ibm.com/developerworks/power/cell/.

[138] USRP: Ettus Research LLC. homepage: http://www.ettus.com/.

[139] KUAR: KU agile radio group web: https://agileradio.ittc.ku.edu/.

[140] M. Robert, C. Aguayo, and P. Balister, "Embedded systems and SDR," *Wireless Symposium Tutorial, Wireless @ VT,* Jun. 7 2006.

[141] V. G. Bose, "Design and implementation of software radios using a general purpose processor," *Ph. d. dissertation, MIT,* 1999.

[142] R. J. Ridder, "Programming digital signal processors with high-level languages," *DSP Engineering Magazine,* Summer 2000.

[143] Python: Programming Language official website: http://www.python.org.

[144] SWIG: Project website: http://www.swig.org.

[145] FFTW: project homepage: http://www.fftw.org/.

[146] ALSA: project homepage: http://www.alsa-project.org/.

[147] wxPython: project homepage: http://www.wxpython.org/what.php.

[148] OSS: project homepage: http://www.opensound.com/oss.html.

[149] D. R. Butenhof, *Programming with POSIX Threads*: Addison-Wesley, 1997.

[150] TUN-TAP: project homepage: http://vtun.sourceforge.net/tun/.

[151] SDRF, "Software defined radio technology for public safety," *Software Defined Radio Forum PS-SIG approved doc 06a_0001_v0.00,* 2006.

[152] SAFECOM, "Statement of requirements for public safety wireless communications and interoperability," SAFECOM Program, Department of Homeland Security Mar. 10 2004.

[153] C. W. Bostian, "A prototype public safety cognitive radio for universal interoperability," Center for Wireless Telecommunications, Wireless Symposium 2006, Wireless at Virginia Tech, Blacksburg, VA.

[154] Java: resource homepage: http://java.sun.com/.
[155] SQL: mySQL open source project homepage: http://www.mysql.com/.

Appendix A: List of Acronyms

ADC	Analog-to-Digital Converter
AGC	Automatic Gain Control
AI	Artificial Intelligence
ALSA	Advanced Linux Sound Architecture
ANN	Artificial Neural Network
API	Application Programming Interface
AWGN	Additive White Gaussian Noise
BER	Bit Error Rate
CBR	Case Based Reasoning
CE	Cognitive Engine
CommTech	Communications Technology
CR	Cognitive Radio
CRKB	Cognitive Radio Knowledge Base
CTVR	Center for Telecommunications Value-chain Research
CWT	Center for Wireless Telecommunications
CWT^2	CWT Cognitive Wireless Technology
DARPA	Defense Advanced Research Projects Agency
DARPA-XG	DARPA NeXt Generation Program
DCA / DSA	Dynamic Channel / Spectrum Allocation
DFT	Discrete Fourier Transform
DSP	Digital Signal Processor
DVB	Digital Video Broadcasting
DySPAN	Dynamic Spectrum Access Networks
EDGE	Enhanced Data rate for GSM Evolution
FCC	Federal Communications Commission
FFT	Fast Fourier Transform
FFTW	Fast Fourier Transform in the West

FGPA	Field Programmable Gate Array
GA	Genetic Algorithm
GPP	General Purpose Processor
GPRS	Generalized Packet Radio Service
GSM	Global System for Mobile Communications
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet Protocol
IRIS	Implementing Radio in Software
K-NN	K-Nearest Neighbor
LLC	Logic Link Control
LO	Local Oscillator
MAC	Media Access Control
MFSK	M-ary Frequency Shift Keying
MIMO	Multiple Input Multiple Output
MLPN	Multi-Layer Perceptron Network
MOGA	Multi-Objective Genetic Algorithm
MPRG	Mobile Portable Radio Research Group
NAO	Notice and Order
NCO	Numerical Controlled Oscillator
NetS	Networking Technology and Systems
NIJ	National Institute of Justice
NPRM	Notice of Proposed Rule Making
NSF	National Science Foundation
OCON	One-Class-One-Network
OS	Operation System
PHY	Physical (communication layer)
PSCR	Public Safety Cognitive Radio
PSD	Power Spectral Density
PTT	Push to Talk
QoS	Quality of Service
QPSK	Quadrature Phase Shift Keying

REM	Radio Environment Map
RRM	Radio Resource Management
RSSI	Received Signal Strength Indication
SCA	Software Communication Architecture
SCD	Spectral Correlation Density
SDR	Software Defined Radio
SDRF	Software Defined Radio Forum
SNR	Signal to Noise Ratio
SQL	Sequential Query Language
SWIG	Simplified Wrapper and Interface Generator
UML	Unified Modeling Language
UMTS	Universal Mobile Telecommunications System
USRP	Universal Software Radio Peripheral
WCDMA	Wide-band Code-Division Multiple Access
WiMAX	Worldwide Interoperability for Microwave Access
WLAN	Wireless Local Area Networks
WRAN	Wireless Regional Area Network
WWRF	Wireless World Research Forum
XML	Extensible Markup Language

Appendix B: PSCR Node System Source Code Structure

PSCR node system software package release v1.0 can be downloaded from CWT web server with authorized access. The download address is:

https://cwt-grad20.cwt.vt.edu/svn/pscr/trunk

From the root folder, the key source code folders are described below:

(1) Trunk/cognitive_engine/

(a) src/sensors/psd

This folder contains both spectrum sweeper and signal classifier. They are compiled with the cognitive engine system by default. Once they are compiled, they can be directly called from Linux shell commands via standard TCP sockets. The supported functionalities and command format is provided by the user manual also included in the PSCR package.

(b) src/xmlscripts

This folder contains all the radio platform definition files and CR system configuration files. They are all written in .xml format. Note that the "gnuradio_hw.xml" is currently called by the cognitive engine to configure and control CWT^2 waveform framework platform. There's also a sensor_psd.xml file that is used to configure the radio environment recognition modules including spectrum sweeper and signal classifier in PSCR node.

(c) src/wsga_fitness_functions

This folder contains all supported fitness functions for specific performance objectives. Although in source code format all the fitness functions are in a single .cpp file. They are compiled to individual dynamic link libraries (DLLs), and can be freely combined and dynamically loaded to guide the GA search for multi-objective solution optimization.

(d) src/genetic_algorithms

This folder contains the GA solution search engine. There are multiple GA engines in this folder for different optimization applications. For PSCR node, the GA engine used is in wsga/ folder. It loads the radio platform definition XML file (in src/xmlscripts) and fitness functions (in src/wsga_fitness_funtions) and search for optimal solutions in terms of waveform configuration and radio platform configuration.

(2) Trunk/CwtGui

This folder contains the complete Java-based PSCR node user interface, and cognitive engine system control module. All the java source codes are in src/ folder, and compiled byte files are in jar/ folder. In src/folder, the main GUI system is coded in GuiWindow.java file; the service data interface is created by GuiTalkMsgReceiver.java and GuiTalkMsgServer.java files. Standard TCP sockets are used for all control and data interfaces between GUI and other PSCR node modules.

(3) Trunk/framework

This folder contains the CWT^2 waveform framework SDR platform. Waveform generation flow graphs are written in .py files. Radio configuration and control interfaces are written in .xml files. Standard XML parsing and radio resource translation is done by xmlParser.py. The top-level waveform framework system is created by framework.py, which includes both control and signal processing hierarchy. Although all MAC algorithms are written in the single mac.py file, they can be dynamically loaded into the framework and control the flow graph. Rate conversion in digital signal processing is managed by pick_bitrate.py file.

(4) Trunk/doc

This folder contains both system installation guide and multiple user manuals for different functional modules of the PSCR node system.

(a) Energy_detection_modulation_classify.doc user manual details how the energy detection and signal classification algorithms are coded, how to use them, and how to continue developing this module.

(b) GUI.doc user manual details the Jave implementation of the user interface. More importantly, it explains the backend information processing and event handling between GUI display front-end and cognitive engine back end. This manual also describes the implementation of Java JDBC API for the cognitive engine solution maker to access knowledge base.

(c) xmlParser_API.doc details the platform independent radio interface provided by CWT^2 waveform framework platform. Standard XML parsing and data format verification are explained. It also briefly introduces the framework configuration and control mechanism and their operation according to the input from radio interface.

(d) Pscr_node_installation_guide.doc provides a detailed step-by-step instruction flow of building up a complete PSCR node system from scratch in a general Linux desktop environment. All the needed external packages are listed and their specific configurations are also provided.

(5) Trunk/tools

This folder contains a set of basic supporting libraries for all the PSCR node modules functionality. This toolbox currently provides three major supporting libraries.

(a) extended_math/ folder provides support of advanced mathematic functions, such as mean, variance, factorial, min, max, sort and so on, as well as defining special data formats like complex vector classes.

(b) tcpip/ folder provides a standard TCP socket creation method in C++, it directly calls the Linux OS kernel threads for maximal efficiency.

(c) xmlparser/ folder provides a standard XML parsing method in C++. It used exPat opensource library to convert the XML data into a dom memory tree.

(6) Trunk/all.sql

This is the image file of the complete PSCR knowledge base implemented in standard mySQL data base. It contains three types of relational tables.

(a) casebase table contains the components for the PSCR solution maker to lookup and build a solution.

(b) Waveform knowledge dictionary contains the specific public safety waveform knowledge for the PSCR radio environment recognition modules to look up and help identify the detected waveform format.

(c) Signal classifier training table contains the training procedure and trained knowledge of the signal classification module. It can be directly loaded to a newly deployed PSCR, so that it can get familiar to the field environment including waveform formats and channel quality immediately instead of going through a long-term self-learning.

(7) Trunk/lanuchall.sh

This the executable file that launches the complete PSCR node. Once it is called, the GUI interface is shown and all the working modes are ready to be activated by the user.

Vita

Bin Le received his B.S. degree in Electrical Engineering from Zhejiang University, Hangzhou, China, in August 2001. Being a top 3% senior undergraduate, Bin was picked by the Shanhai Bell Co. Ltd to develop broadband access network. After graduation, he worked in Bell on IP-switching core development for one year. In 2002, Bin came to Virginia Tech, and joined Center for Wireless Telecommunications (CWT) as a M. S. graduate student advised by Dr. Charles W. Bostian, where he conducted the research on the direct-conversion receiver (DCR) and supporting radio frequency integrated circuit (RFIC) technologies. In 2004, Bin was approved to become a Direct Ph.D. student, and started the research on Cognitive Radio technology. In early 2005 he founded CWT Cognitive Wireless Technology (CWT2) research thread focusing on the development of the cognitive radio node system as a general solution for different cognitive network applications. From August 2006, more students joined CWT2 team. Bin led the team going through the development from the lab testbed to the final prototype ready for field testing.

Bin's research interest includes cognitive radio and networking, cross-layer optimizations, software defined radios, artificial neural networks and evolutionary algorithms. He has published multiple book chapters, journal and magazine papers, and several tens of conference papers. He currently owns one intellectual property, and is applying another two at Virginia Tech.