

AFRL-IF-RS-TR-2005-281
Final Technical Report
August 2005



NEXT GENERATION (XG) ARCHITECTURE AND PROTOCOL DEVELOPMENT (XAP)

BBN Technologies

Sponsored by
Defense Advanced Research Projects Agency
DARPA Order No. P858

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the U.S. Government.

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

STINFO FINAL REPORT

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

AFRL-IF-RS-TR-2005-281 has been reviewed and is approved for publication

APPROVED: /s/

RICHARD D. HINMAN
Project Engineer

FOR THE DIRECTOR: /s/

WARREN H. DEBANY, JR., Technical Advisor
Information Grid Division
Information Directorate

REPORT DOCUMENTATION PAGE			<i>Form Approved</i> OMB No. 074-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing this collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503				
1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE AUGUST 2005	3. REPORT TYPE AND DATES COVERED Final Jun 03 – Dec 04	
4. TITLE AND SUBTITLE NEXT GENERATION (XG) ARCHITECTURE AND PROTOCOL DEVELOPMENT (XAP)			5. FUNDING NUMBERS C - F30602-03-C-0139 PE - 63760E PR - P858 TA - XG WU - II	
6. AUTHOR(S) Ram Ramanathan and Craig Partridge				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) BBN Technologies 10 Moulton Street Cambridge Massachusetts 02138			8. PERFORMING ORGANIZATION REPORT NUMBER N/A	
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Defense Advanced Research Projects Agency AFRL/IFGC 3701 North Fairfax Drive 525 Brooks Road Arlington Virginia 22203-1714 Rome New York 13441-4505			10. SPONSORING / MONITORING AGENCY REPORT NUMBER AFRL-IF-RS-TR-2005-281	
11. SUPPLEMENTARY NOTES AFRL Project Engineer: Righard D. Hinman/IFGC/(315) 330-3616/ Richard.Hinman@rl.af.mil				
12a. DISTRIBUTION / AVAILABILITY STATEMENT APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.				12b. DISTRIBUTION CODE
13. ABSTRACT (Maximum 200 Words) The DARPA XG program seeks to develop and demonstrate adaptive spectrum utilization techniques which can be used to access unused spectrum without interference to incumbent users. This report describes the progress which was made to develop an XG architecture which is capable of accommodating a wide range of adaptive protocols operating within the bounds of spectral policy. Additionally this development included the definition of a machine readable policy language and a policy conformance reasoner to validate if candidate transmissions are compliant to spectrum policy.				
14. SUBJECT TERMS Adaptive Communication, Cognitive Radio, XG, Adaptive Spectrum Utilization, Machine Readable Policy				15. NUMBER OF PAGES 248
				16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT UL	

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18
298-102

Table of Contents

1	SCOPE	1
2	BACKGROUND	2
2.1	THE XG VISION OF POLICY AGILITY	3
2.2	THE XG ARCHITECTURAL FRAMEWORK	4
3	POLICY LANGUAGE FRAMEWORK	6
3.1	CONCEPT OF OPERATIONS	7
3.2	LANGUAGE OVERVIEW	11
3.3	POLICY LANGUAGE REPRESENTATION REQUIREMENTS	14
3.4	POLICY ONTOLOGIES	17
3.5	POLICY CONFORMANCE REASONER (PCR) SOFTWARE AND POLICY UTILITIES	18
3.6	ENCODED POLICY EXAMPLES AND PCR TESTS	21
4	ABSTRACT BEHAVIORS	26
4.1	AN ABSTRACT MODEL OF THE XG RADIO SYSTEM	27
4.2	XG INTERFACES	30
4.3	XG BEHAVIORS	32
4.4	XG INFORMATION OBJECTS	34
4.5	XG SYSTEM INTERACTIONS	35
5	MODELING AND SIMULATION OF PROTOCOLS FOR XG	38
5.1	XG EVALUATION PLATFORM (XEP)	38
5.2	PROTOCOLS FOR OPPORTUNISTIC SPECTRUM ACCESS	39
5.3	SIMULATION RESULTS	41
6	TECHNICAL INTERACTIONS	46
6.1	WORKING GROUP MEETINGS AND OTHER BRIEFINGS TO XG PERFORMERS	46
6.2	WEBSITE AND MAILING LISTS	46
6.3	EXTERNAL BRIEFINGS AND INTERACTIONS	46
6.4	ACADEMIC COLLABORATIONS	47
7	SUMMARY AND FUTURE WORK	48
7.1	BBN'S ACCOMPLISHMENTS IN THE XG PROGRAM	48
7.2	FUTURE WORK	50
	References	52
	APPENDIX A XG VISION REQUEST FOR COMMENTS	53
	APPENDIX B XG ARCHITECTURAL FRAMEWORK	69
	APPENDIX C XG ABSTRACT BEHAVIORS REQUEST FOR COMMENTS	86
	APPENDIX D XG POLICY LANGUAGE FRAMEWORK	151

List of Figures

FIGURE 1: MACHINE UNDERSTANDABLE POLICIES. WITH THIS, CHANGING THE POLICY MERELY REQUIRES LOADING A DIFFERENT FLASH CARD OR DOWNLOADING ANEW.	4
FIGURE 2: THIS ILLUSTRATES THE DECOUPLING OF POLICIES, BEHAVIORS, AND PROTOCOLS, ALONG WITH TRACEABILITY AND THE FOUR COMPONENTS OF OUR FRAMEWORK.	5
FIGURE 3: MACHINE UNDERSTANDABLE POLICIES ARE NECESSARY TO EXPLOIT THE EMERGING AGILITY OF DEVICES AND ENABLE IN-SITU POLICY-BASED CONTROL OF RADIO BEHAVIORS	7
FIGURE 4: POLICY LANGUAGE ACTORS.....	8
FIGURE 5: POLICY-AGILE OPERATION OF XG SPECTRUM-AGILE RADIO.....	10
FIGURE 6: STRUCTURE OF POLICY FACTS	12
FIGURE 7: POLICY CONFORMANCE REASONER SOFTWARE ARCHITECTURE	19
FIGURE 8: POLICY E1 WITH PSD CONSTRAINTS MODIFIED BY E2 TO INTRODUCE A NOTCH-OUT BAND	22
FIGURE 9: VALIDATION PERFORMANCE OF THE POLICY CONFORMANCE REASONER	25
FIGURE 10: XG RADIO SYSTEM ABSTRACT MODEL	27
FIGURE 11: XG INTERFACES.....	30
FIGURE 12: XG INTERNAL AND PROTOCOL BEHAVIOR CLASSES.....	33
FIGURE 13: XG INFORMATION OBJECTS AND THEIR ASSOCIATIONS	35
FIGURE 14: SPECTRUM AWARENESS IS GAINED THROUGH SENSING AND PROTOCOL BEHAVIORS	36
FIGURE 15: THE SYSTEM STRATEGY REASONER IDENTIFIES AND ALLOCATES OPPORTUNITY INSTANCES FOR THE RADIO; IT MAY ADDITIONALLY ACCESS RADIO PRIMITIVES TO MODIFY SYSTEM BEHAVIOR	36
FIGURE 16: OPPORTUNITY INSTANCES ARE ALLOCATED, VALIDATED AGAINST POLICY, THEN USED AFTER ASSERTING THAT ALL PARAMETER VALUES ARE SET APPROPRIATELY AND AFTER COORDINATION WITH PEERS	37
FIGURE 17: XG EVALUATION PLATFORM SYSTEM ARCHITECTURE	38
FIGURE 18: IDLE CHANNEL SELECTION FROM HOLE INFORMATION ARRAYS.....	40
FIGURE 19: SPECTRUM UTILIZATION BY LEGACY (UPPER LEFT) AND XG (LOWER LEFT) SYSTEMS GIVEN THE PRIMARY SYSTEM UTILIZATION IS 40%. A MORE DETAILED SNAPSHOT OF THE SPECTRUM UTILIZATION BY THE XG SYSTEM IS SHOWN ON THE RIGHT. BANDWIDTH IS MEASURED IN TERMS OF NUMBER OF 100kHz CHANNELS USED.....	42
FIGURE 20: COMPARING THROUGHPUT ACHIEVED BY XG AND LEGACY SYSTEMS. THE MAXIMUM ACHIEVABLE USING A THEORETICAL BOUND (BASED ON SHANNON CAPACITY), AND THE MAXIMUM ACHIEVABLE IF ONLY QPSK MODULATION IS ALLOWED ARE ALSO SHOWN.....	44
FIGURE 21: XG POLICY TECHNOLOGY ROADMAP.....	49
FIGURE 22: XG SYSTEM INTEGRATION ARCHITECTURE	50

List of Tables

TABLE 1: OPPORTUNITY SEARCH BASED ON NON-LINEAR REAL CONSTRAINT SOLVING	22
TABLE 2: POLICY TOOLS CAPABILITIES SUMMARY	24
TABLE 3: DEFAULT SIMULATION PARAMETERS	42

1 Scope

This is a report on the *XG Architecture and Protocol Development (XAP)* project. The XAP project was funded under the DARPA neXt Generation (XG) Communications program (Phase 2), beginning in June 2003, and ending in December 2004. XAP is a follow on to the *Medium Access Control Protocols for XG (X-MAC)* project that was funded under the DARPA XG program (Phase 1).

This report contains an overview of the project accomplishments. We emphasize that this report is only an overview, and details can be found in a number of supporting documents and Request for Comments (RFCs) contained in the XAP distributions, including:

1. The *XG Vision RFC* [XGV], which describes the vision and goals of the XG program in general and X-MAC in particular.
2. The *XG Architecture RFC* [XGAF], which presents the architecture, system components, and a high level concept of operations for XG communications.
3. The *XG Abstract Behaviors RFC* [XGAB], which identifies key behaviors that must be implemented by an XG system, organizes them, and describes the behaviors.
4. The *XG Policy Language Framework RFC* [XGPLF], which describes the policy specification meta-language for implementing machine-understandable policies.
5. The *XG Policy Conformance Reasoner* software and documentation [PCR], which describes a reference implementation based on the XG Policy Language Framework RFC v1.0.
6. The *XG Evaluation Platform* [XEP], which describes the OPNET models of XG protocols and simulation results.

Accomplishments under XAP can be classified into two broad categories – (a) the development of a framework for managing the key aspects of radio behavior through flexible application of policies, (b) design, modeling and simulation of key protocols for opportunistic spectrum access. Items (1)-(5) above fall under the framework, and (6) under the protocol modeling and simulation. For most of the project duration, these two sets of activities were conducted in parallel.

The remainder of this document is organized as follows. In Section 2, we present the background and motivation for XAP. Further, we describe the XG vision and architectural framework that was developed largely in the earlier XMAC contract, and has been refined during the XAP effort through the working group process. In the XAP effort, we have developed specific technologies that can enable policy-agile opportunistic spectrum access by XG systems. These include the policy language framework and policy conformance reasoner software that we describe in Section 3, the abstract behavior specification that we describe in Section 4, and the XG evaluation platform and protocol simulation results that we describe in Section 5. In Section 6 we discuss technical interactions related to the XG program both within the XG program including working group activities as well as external interactions including briefings to government, academia and industry. In Section 7, we conclude with a summary of lessons learned from the project, a roadmap for the XG policy-agile opportunistic spectrum access technology (status at the end of Phase 2 and projections for Phase 3 and beyond), and recommendations for future work.

2 Background

There are two significant problems confronting wireless communications with respect to spectrum use:

- ◆ *Scarcity*. The current method of allotting spectrum provides each new service with its own fixed block of spectrum. Since the amount of useable spectrum is finite, as more services are added, there will come a point at which spectrum is no longer available for allotment. We are nearing such a time, especially due to a recent dramatic increase in spectrum-based services and devices.
- ◆ *Deployment difficulty*. Currently, extensive, frequency-by-frequency, system-by-system coordination is required for each country in which these systems will be operated. As the number, size, and complexity of operations increase, the time for deployment is becoming unacceptably long.

Both problems are a consequence of the centralized, static nature of current spectrum allotment policy. This approach lacks the flexibility to aggressively exploit the possibilities for dynamic reuse of allocated spectrum over space and time, resulting in very poor utilization and *apparent* scarcity. It also mandates a priori assignment of spectrum to services before deployment, making deployment difficult.

Preliminary data indicates that large portions of allotted spectrum are unused (refer the Spectrum Policy Task Force report [SPTF]). This is true both spatially and temporally. That is, there are a number of instances of assigned spectrum that is used only in certain geographical areas, and a number of instances of assigned spectrum that is only used for brief periods of time. This wastage of assigned spectrum is bound to increase in future – spatially, due to the increasing localization of propagation due to radio devices using higher frequencies, and temporally due to the proliferation of services that are highly bursty in nature.

Studies have determined that even a very straightforward reuse of such “wasted” spectrum can provide an order of magnitude improvement in available capacity. It can be concluded that the issue is not so much that spectrum is scarce, but that we do not have the technology to effectively manage access to it in a manner that would satisfy the concerns of current licensed spectrum users.

XG (neXt Generation Communications) is a DARPA-funded research program based on the (now generally accepted) premise described above that the historic (and current) method of authorizing spectrum use—static, administrative allocation—results in an apparent scarcity of spectrum that can be avoided by the proper application of dynamic spectrum sharing techniques. The goals of the XG program are:

1. Demonstrate through technological innovation the ability to utilize available (unused, as opposed to unallocated) spectrum more efficiently.
2. Develop the underlying architecture and framework required to enable the practical application of such technological advances.

The XG program had several performers, including BBN. The goal of the other performers was to develop opportunistic spectrum access technologies. BBN’s role within the program was unique – we served as the developer of a framework within which such technologies could evolve.

Specifically, XAP had two main technical goals (and accomplishments):

1. *Develop a long-lived framework for managing the key aspects of radio behavior through flexible application of policies*. In order that the radio is policy-agile, we require a framework in which policies are written in a way that can be interpreted by the radio, and the radio is able to exploit such expression of policies. Furthermore we require a specification of key behaviors of an XG system so as to provide a common set of abstractions for policy administrators and XG system designers.

2. *Develop protocols for XG communications and analyze them using modeling and simulation.* The development of protocols gives insight into the right framework, and modeling captures the performance variations as a function of various system and environmental parameters. Further, our simulation system serves as a “proxy XG radio” in the framework.

BBN Technologies also coordinated an XG Working Group comprised of Government and support staff, and the XG Phase 2 program participants.

As part of the second goal the XG program, participants (known collectively as the XG Working Group or XG-WG) produced a series of Requests for Comments (RFC) that together describe the proposed XG architecture and framework. These documents are RFCs because the authors recognize that the final development of such an important technology cannot be accomplished by a small group of individuals. It requires input and participation from a broad representation of the affected community. Thus it is hoped these RFCs will spur that community to provide feedback in order to assure an organized and technically valid approach to the evolution of this architecture. BBN was the primary developer of all the RFCs to date, with inputs from DARPA, AFRL, and the other XG performers.

The XG Vision RFC provides the motivation and scope of dynamic spectrum sharing envisioned by the XG program and describes an approach for developing the XG architecture. It is highly recommended that the XG Vision RFC be read before reading the others. The XG Architectural Framework RFC presents the XG architecture, system components, and a high level concept of operations for XG communications. The XG Policy Language Framework was the third to be released. The draft XG Abstract Behaviors RFC is being released (to the XG program participants) at the same time as this report. It provides an abstract view of the operation of XG radios, so that policies may be written to govern the operation of radios that are yet to be engineered.

In order to address the scarcity and deployment difficulty problems, XG is pursuing an approach wherein static allotment of spectrum is complemented by the opportunistic use of unused spectrum on an instant-by-instant basis, in a manner that limits interference to primary¹ users. In other words, the basic concept of operation is as follows: a device first “senses” the spectrum it wishes to use and characterizes the presence, if any, of primary users. Based on that information, and regulatory policies applicable to that spectrum, the device identifies spectrum opportunities (in frequency, time, or even code), and transmits in a manner that limits (according to policy) the level of interference perceived by primary users. We term this approach *opportunistic spectrum access*.

Opportunistic spectrum access also provides far easier deployment, or rapid entry, into regions where spectrum has not been assigned. Only minimal prior coordination is necessary, greatly easing the restrictions to meet the deconfliction requirements. This is helpful both in civilian applications such as the entry of a wireless LAN technology in less developed regions, and in military operations requiring high tempo and quick reaction time.

2.1 The XG Vision of Policy Agility

The realization of opportunistic spectrum access is highly challenging. Several problems must be solved: sensing over a wide frequency band; identifying the presence of primaries and characterizing available opportunities; communication among devices to coordinate use of identified opportunities; and most importantly, definition and application of interference-limiting policies, and utilization of the opportunities while adhering to such policies.

The true potential of this new approach can be exploited only if in addition to spectrum agility, we provide *policy agility* – that is, a way by which the policies controlling the behavior can be dynamically

¹ Users that are licensed to use the spectrum in question, subject to regulatory constraints are called primary users.

changed. That is, policies are not embedded in the radio, but can be loaded “on-the-fly”. Policy agility allows adaptation to policies changing over time and geography. Further, technology (spectrum agility) can be developed in advance of policies. This is important for breaking the chicken-and-egg dilemma that exists today, where regulatory bodies must wait for technology before drafting policies and technology must wait to see what the policies will look like.

The use of policy agility using *machine understandable* policies is depicted in Figure 1. Starting from the left, spectrum policies are encoded in a machine interpretable form and loaded into the XG device. The XG device then operates in accordance with its interpretation of these policies. Policies may be loaded using smart media or over the Internet. In order to change the policies we simply need to load a new version. For instance, operating in a different country would require merely downloading from a different website or new smart card.

Although recent years have seen some of the components for opportunistic spectrum access mature (e.g. software radios), we are a long way from a prototypical system. Further, no work exists in the area of decoupling the policies from the implementation. This yawning gap between current state-of-the-art and what is required for opportunistic spectrum access is the motivation behind XG.

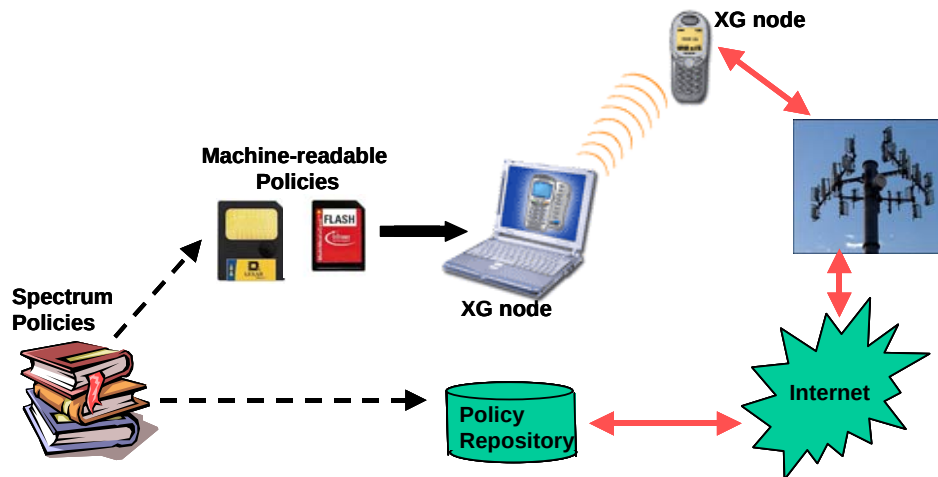


Figure 1: Machine understandable policies. With this, changing the policy merely requires loading a different flash card or downloading anew.

A more comprehensive discussion of the XG vision and motivation can be found in the XG Vision RFC [XGV].

2.2 The XG Architectural Framework

The XG framework is based on a decoupling of three fundamental elements in the design of an XG system: *policies, behaviors, and protocols*. By this, we mean that policies, behaviors and protocols are developed separately with some kind of “connections” defined between them. This concept is illustrated in Figure 2.

Decoupling allows adaptation to policies that vary over time and geography. Technology can be developed in advance of policies, and worldwide deployment would be greatly simplified. Furthermore, sub-policy management, as required for secondary spectrum markets, is easier. Policies no longer have to reflect the common denominator of competing technologies, and can be tailored to the diverse system capabilities expected for opportunistic spectrum access. Decoupling behaviors from protocols allows us to control *what* needs to be done separately from *how* it is implemented, resulting in a cleaner and more flexible architecture. Indeed, we argue that a decoupling approach is not just beneficial but pretty much a requirement for harnessing the full potential of opportunistic spectrum access.

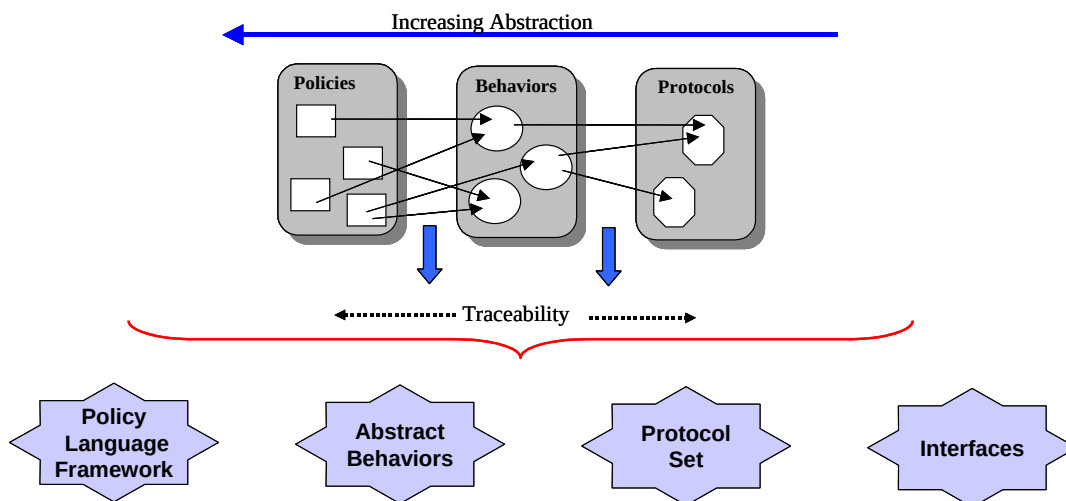


Figure 2: This illustrates the decoupling of policies, behaviors, and protocols, along with traceability and the four components of our framework.

A central notion in this approach, and a notion that is enabled by this approach, is that of *traceability*. Behaviors, preferably one or more of a core set of abstract behaviors, should be traceable to policies. This provides two advantages: first, it helps make the verification of new policies easier, and second, when an XG radio is deployed in a new region, it is easier to affirm that the XG system will behave in a certain way. It allows us to accredit based on ability to correctly interpret and implement policies.

As shown in the figure, the task of decoupling policies, behaviors and protocols requires the definition of four components in the framework: policy language, abstract behaviors, protocols, and interfaces. We describe below our accomplishments on the definition of the policy language, abstract behaviors, and one interface – namely the policy language interface. Protocols were only developed as part of the parallel modeling and simulation effort within BBN (that is, unlike the language, behaviors and policy interface, this did not have program-wide scope and consequently did not involve participation of the XG working group members, namely, the Government and performers). We give a very brief overview of some of these protocols in Section 5.

We reiterate that this report is only a summary. The reader should refer to the particular RFC (referenced in the respective section) for details. We also note that the development of the framework is an evolving process and this report reflects the current snapshot in that evolution. The follow on projects in Phase 3 will further evolve the framework, and indeed, may modify some of the definitions given here.

The XG Vision and Architectural Framework summarized in this section were largely developed within the earlier BBN XMAC effort in Phase 1 of the XG program. These were specified in the form of two RFCs [XGV, XGAF], which we have refined during the XAP effort through the working group process.

The XAP effort builds upon and significantly extends the work from the XMAC effort. We have developed specific technologies to enable policy-agile opportunistic spectrum access by XG systems. These include the policy language framework (policy ontologies and the policy conformance reasoner software) [XGPLF], the abstract behavior specification [XGAB], and protocol simulation results [XEP].

3 Policy Language Framework

XG-enabled radios will be able to utilize available spectrum intelligently based on knowledge of actual conditions rather than using current conservative spectrum management methods (static spectrum assignments). In this way, XG technologies will utilize spectrum in a much more efficient manner than can be done today. Furthermore, without the need to statically allocate spectrum for each use, networks can be deployed much more rapidly. Today the military, for example, must make spectrum use requests to their spectrum managers, who must deconflict them, and make static assignments far in advance of deployment. With XG capabilities, this process can be significantly shortened or perhaps even eliminated beyond decisions made at the highest level.

Radios must adhere to rules that apply to their operation. A major intent of such rules is to avoid or reduce interference among users. Such rules may cover both transmission and reception functions of a radio. Currently such rules are enumerated in spectrum policy as produced by various spectrum authorities such as the FCC or the NTIA in the U.S.A. In this document we use the term *spectrum policy* to refer to any externally (to the radio) imposed rules for spectrum use.

A radio that is capable of dynamically utilizing spectrum must be able to adhere to rules corresponding to the many uses of which it is capable—not just one use as with most current radios. XG radios will be expected to operate over a wide range of frequencies and within different geopolitical regions. Therefore, they must incorporate a real-time adaptive mechanism for conforming to the policies applicable to each situation. In other words, XG radios must be *policy-agile*, by which we mean both that the radios are situationally adaptive to the current policy, and that they allow policy to be dynamically updated as well.

Spectrum policies relevant to a given radio may vary in several ways:

1. Policies may be altered over time.
2. The radio (along with its user) may move from one policy administrative domain to another (e.g. a military user may be deployed to a different country).
3. Policies may be dependent on time of day or year.
4. A spectrum owner/leaser may impose policies that are more stringent than those imposed by a regulatory authority.
5. The spectrum access privileges of the radio may change in response to a change in radio user.

To be truly versatile, the XG radios should be responsive to such changes in policy.

Current radios support only a small number of modes of operation and a limited range of intended operating environments. With only limited hardware agility, all the relevant policy sets that apply can be hard-coded into the radio. However, with the increasing agility and programmability of radio hardware (as illustrated in Figure 3), especially when combined with the prospect of opportunistic spectrum sharing, both the number of modes of operation as well as the range of operating environments for the radio will increase tremendously. As a result, the number of different policy sets that apply to these various modes and environments will grow in a combinatorial fashion. This will make it impractical to hard-code into the radio discrete policy sets to cover every case of interest. The accreditation of each discrete policy set would also be a major challenge. In the case of software-defined radios, it would require the maintenance of downloadable copies of software implementations of each policy set for every radio platform of interest.

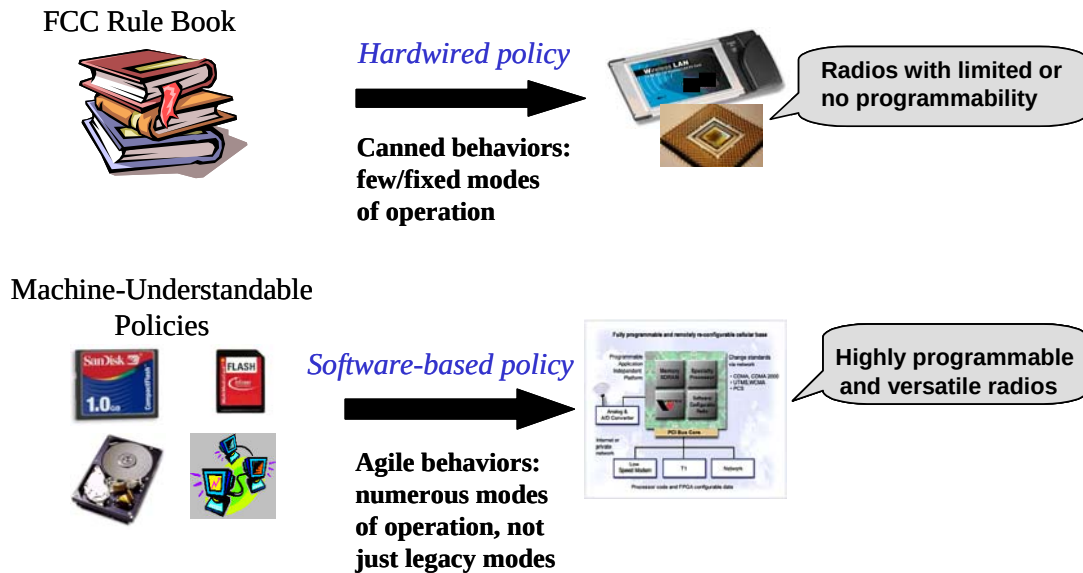


Figure 3: Machine understandable policies are necessary to exploit the emerging agility of devices and enable in-situ policy-based control of radio behaviors

We need a more scalable way to express and enforce policy. The complexity of accrediting policy conformance for XG radios and the desire for policy agility lead to the conclusion that XG radios must be able to read and interpret policy. We must, therefore, be able to express policy in a well-defined language framework.

At this point we note the distinction between the *policy language* and the *policy language framework*. We use the term *XG policy language* to refer to the language elements (including the syntactic and semantic structure of the language, and the domain-specific keywords), as well as the external (lexical) representation of the language. Overall, in this document we emphasize the language elements; we defer the lexical representation to the appendices. For this purpose, we make use of a shorthand notation to illustrate the concepts. We use the term *policy language framework* to include the policy language itself, the concept of operations of creation and use of machine understandable policies, and the computational logic, tools, and techniques for policy processing.

It is important not to confuse the policy language with its lexical representation or its syntax. We will use OWL representation (based on RDF and XML) as a basis of expression—these are not to be confused with the XG policy language. Rather, the XG policy language consists of terms (along with their precise meanings), interrelationships between the terms, some constructs and idioms that are required to express policy, and the mechanisms by which the language can be extended.

In Section 3.1 we describe a concept of operations—both how machine understandable policies are created and encoded by policy administrators, as well as how the encoded policy is used by policy-agile radios. In Section 3.2, we provide a brief overview of the features of the policy language and the policy processing logic. In Section 3.3, we examine the requirements for policy language representations and we propose an extensible, standards-based machine-understandable representation for policies.

3.1 Concept of Operations

The concept of operations of the XG policy language framework describes how machine understandable policies are created and how they are used. First we will describe the actors involved,

their roles in creating and using policy, and the elements of the policy that they use or control. Then we describe how radios can interpret and use the encoded policy.

3.1.1 Development of Machine Understandable Policies

Three different types of actors are involved in the definition and use of the policy language and the policy rules: language designers, policy administrators, and spectrum users. These actors will interact with the policy language and policy rules using tools appropriate for their role as illustrated in Figure 4 and described below.

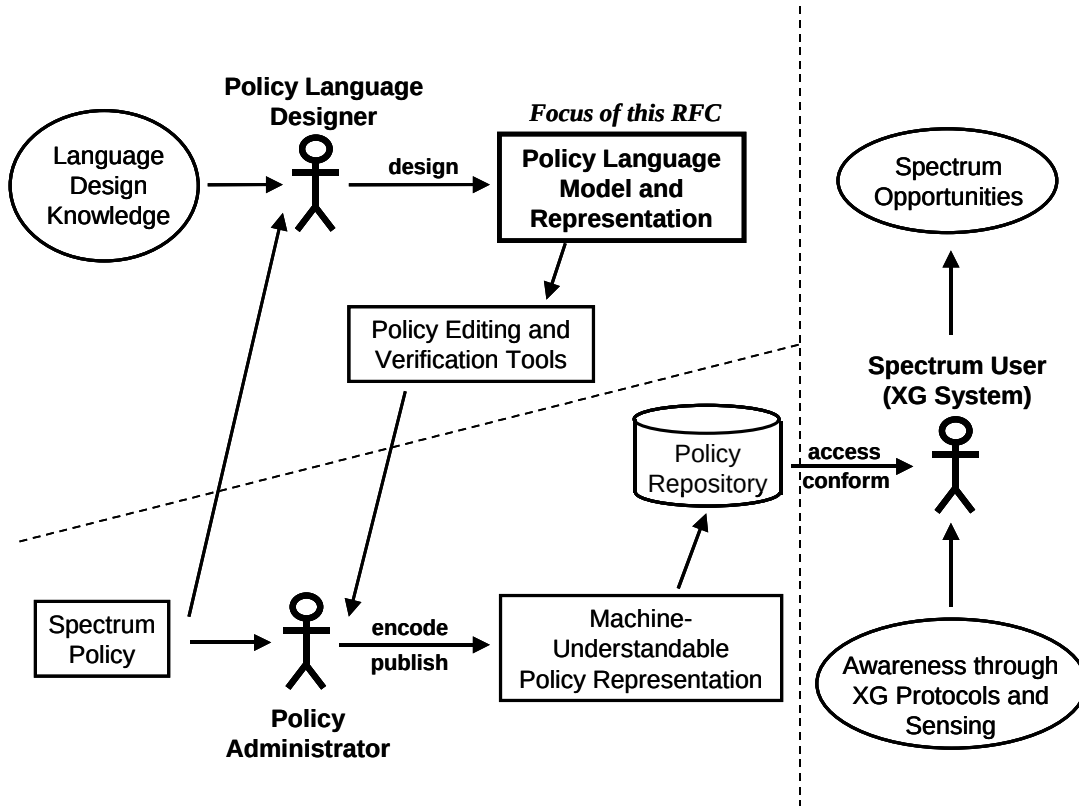


Figure 4: Policy Language Actors

Language designers create the *language model*, which defines the high-level objects of the language along with their semantics. This document describes an initial version of the language model. We expect that eventually a standards committee will draft future versions of the model. The language model is published at a well-known URI for policy administrators and policy users to access. The language designer may utilize tools (such as UML modeling tools, tools to convert UML to a machine-understandable language and other visualization tools yet to be developed) in order to facilitate the design process.

A *policy administrator* (who is not necessarily the policy maker) is responsible for developing and encoding spectrum policy using the policy language produced by the language designers. The policy administrator makes the policy available to spectrum users. Administrators do not have to know all the details of the language presented here, as they will likely encode the policy by using a (perhaps graphical) tool, called an instance editor, which hides the notational complexity of the language. The administrator also uses analysis tools to verify in advance that the encoded policy will have the desired effect and is consistent with existing policies.

In the short term, policy may be described first in English using engineering and legal terms that are commonly understood by the community of interest. Policy administrators will likely continue to use established procedures to interpret spectrum policy prior to encoding.

Given a policy expressed in English, the first step is to perform an analysis to determine how to structure the policy in a form best suited for machine understanding. Once this analysis is performed and the elements of the policy are mapped to the elements of the XG policy language, the policy can then be readily encoded in the XG policy language (XGPL). BBN has developed conversion tools that enable the use of a shorthand notation. Other tools to validate the syntactic and logical correctness of the policy represented are the subjects of future work. As this technology matures, we anticipate that a number of graphical and scripting tools will be developed to make the process easier.

A *spectrum user*, such as an XG system, must be able to use the policy to assess whether policy allows identified potential spectrum opportunities and to understand the constraints the policy administrators have placed on their use. The next section details how the spectrum user uses the policy.

3.1.2 Policy Usage

In this section, we describe a concept of operations for how a policy-agile spectrum user, such as an XG radio system, can use machine-understandable policy. A radio can use the machine-understandable policies to assure that its use of spectrum conforms to policy, as well as to modify radio behaviors in order to identify and utilize spectrum opportunities that are authorized. In this section, we will focus on the former, namely, how to assure policy conformance. The draft XG Abstract Behaviors RFC [XGAB] describes XG radio behaviors related to acquiring and sharing spectrum awareness, as well as behaviors related to identifying, selecting, coordinating, and using available spectrum opportunities as authorized by policy.

Figure 5 is a logical functional block diagram of the policy usage concept of operations. The placement of these functional blocks in hardware or software is to be determined by the designers of each individual system.

There are four main components in this functional decomposition:

- **Sensor:** provides situational information to the radio about the spectral environment at a given location and time. This information is key to being situationally aware of spectrum opportunities enabled by policy. We note that the sensor outputs need not be limited to the RF spectral environment; the outputs could include a variety of other information (e.g. geo-location, temperature, and proximity to specific targets) that could be used as parameters for system policy.
- **Radio Platform:** provides the basic hardware and primitives of a host radio system that enables opportunistic use of spectrum (e.g., RF front end, DSP hardware, system software including the OS, middleware, and libraries, and primitives for networking protocols, waveform agility, and beam forming).
- **Policy Conformance Reasoner:** manages accredited policy information, which includes interpreting the policy language, and reasoning based on accredited policy (e.g., approved by a regulatory authority) and related background knowledge to determine whether the proposed spectrum use is policy-conformant or not; this key component to ensuring policy conformance is largely system independent and does not tell the radio platform or the system strategy reasoner what to do.
- **System Strategy Reasoner:** determines the system's strategy for opportunistic spectrum sharing under regulatory and system policy constraints; this reasoner is aware of system-

specific optimizations and tradeoffs and has control over the radio platform. In Figure 5 we show the system strategy reasoner as logically separated from the rest of the radio. This highlights the potential for reusability.

We envision that the radio platform includes an accredited kernel, within which it implements (grounds) accredited parameters and processes governed by policy. We further envision that only the policy conformance reasoner, the sensor, and the accredited kernel of the radio platform are within the accreditation boundary (shaded portions in Figure 5). Parts of the system that lie within the accreditation boundary are both necessary and sufficient to ensure policy conformance. As per the XG vision [XGV], system and protocol innovations including the system strategy reasoner are outside the accreditation boundary.

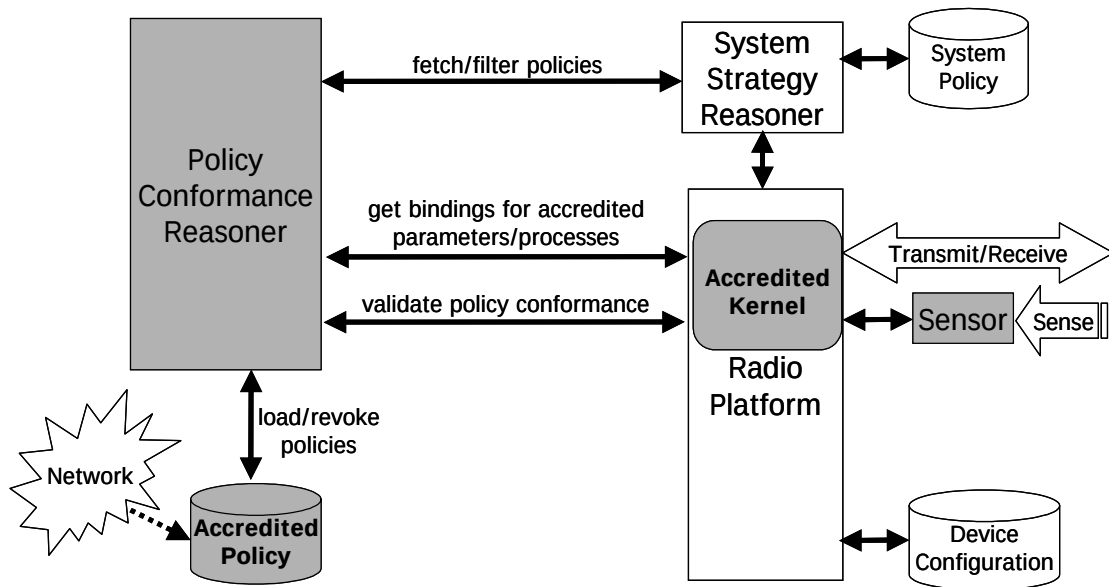


Figure 5: Policy-Agile Operation of XG Spectrum-Agile Radio

This architectural separation of the regulatory policy conformance function from the system dependent optimizations and tradeoffs allows the policy conformance reasoner to be reused and eases accreditation. Rather than separately accredit each of n radio configurations for each of m policy sets for a total of $m \times n$ operations, this approach reduces the required number of operations to $m+n+1$. The policy conformance reasoner must be accredited once, each policy set once, and each radio configuration once.

We describe the functions and interactions of the policy conformance reasoner in more detail here. The policy conformance reasoner performs three basic services:

- First, it loads (and possibly revokes) and interprets accredited policy instances, such as policies specified by a regulatory body, which are represented using the machine-understandable policy language.
- Second, it makes use of the policy structure to respond to queries (by a system strategy reasoner) to filter policies based on selection criteria (e.g. to a specified radio, or intended operating environment).
- Third, it determines whether or not the spectrum use proposed by the radio platform conforms to policy.

In order to explain the operation of the conformance reasoner, we provide a brief overview of the structure of policy rules in the XG policy language. Each policy rule is composed of three parts: a

selector description, an opportunity description, and a usage constraint description. A *description* is a Boolean predicate expression that involves regulated parameters and methods (for which bindings and groundings are provided by the radio platform) and additional policy parameters (for which bindings are provided within the policy itself). An *instance* is a set of bindings that can be used to determine the truth-value of a *description*.

A policy is *selected* and applied if the proposed selector instance satisfies the selector description. If the proposed opportunity instance matches the corresponding opportunity description, this provides *authorization* for the opportunity². The proposed usage instance must fulfill certain *obligations*, i.e., it must satisfy the corresponding usage constraint descriptions. The use of selection, authorization, and obligation to structure policy rules is a paradigm that is used in other policy systems as well (e.g., KAoS).

We now present a notional sequence of operations using the functional decomposition presented in Figure 5. After loading any configuration information, the radio platform in conjunction with the sensor (potentially using a sensing strategy determined by system strategy reasoner) acquires awareness of its situation. The system strategy reasoner has access to the configuration, state, and awareness acquired by the radio platform through means that are specific to the system implementation. Based on system and regulatory policy and knowledge of the radio platform, the system strategy reasoner enables the radio platform to identify and characterize available opportunities and a suitable use of those opportunities. The characterization can result in, for example, binding values to relevant parameters.

Once an opportunity is identified and its intended use is characterized as a set of selector, opportunity, and usage constraint instances, the radio platform must present this set to the policy conformance reasoner for validation. In order to validate the instances, the policy conformance reasoner may need to obtain parameter bindings and invoke method groundings implemented within the accredited kernel of the radio platform. If the requested set of instances is validated, the XG radio may then use them to transmit.

The system strategy reasoner has the function of influencing radio behaviors in response to policy (and situational knowledge) in order to identify and utilize available spectrum as authorized. We note that there is significant scope for design diversity, innovation, and optimization within the system strategy reasoner, as well a potential for its reuse across several radio platforms. A simple system strategy reasoner may try only a limited range of opportunities and uses known to work with the radio and typical policies. A more sophisticated system strategy reasoner can include dynamic constraint solving capabilities; for example, it can query the policy conformance reasoner for applicable policy constraints (based on the state of the environment and the capabilities of the radio platform), and then determine a strategy to traverse the policy decision space efficiently to find good opportunities for the radio platform to use.

The design and specification of the interfaces between the radio platform and the policy conformance reasoner is a subject for future work.

3.2 Language Overview

This section provides a high-level overview of the XG policy language (XGPL) including the language elements, and the policy processing logic. In this section we also present the requirements that motivate the use of OWL as the machine-understandable representation for the XG policy language.

²Positive and negative authorizations are supported.

3.2.1 Language Elements

An XG policy instance consists of a set of facts and rules. Facts define the policy and rules describe how to process the facts and hence how the facts relate to one another.

A policy rule is a statement of policy consisting of a set of facts, but not the rules for interpreting those facts. All other facts either further explain a policy rule or make statements about a policy rule. Policy rules encode statements of policy, such as, “if peak received power is less than -80dBm then maximum EIRP is 10mW.” Most other facts either support the policy rules or refer to the policy rules. A policy rule links three facts: a selector description, an opportunity description, and a usage constraint description to describe a single statement in a policy, as illustrated in Figure 6.

The first fact in a policy rule is a selector description. This fact is used to filter policy rules to the set of rules that may apply to a given situation. The selector description points to one or more facts describing the authority that has jurisdiction over the policy, the frequency, time, and region the policy covers, and a description of the radio to which the policy rule applies. For example, a selector description may include filters such as “applies to operation in England” or “applies to operations in the broadcast bands”.

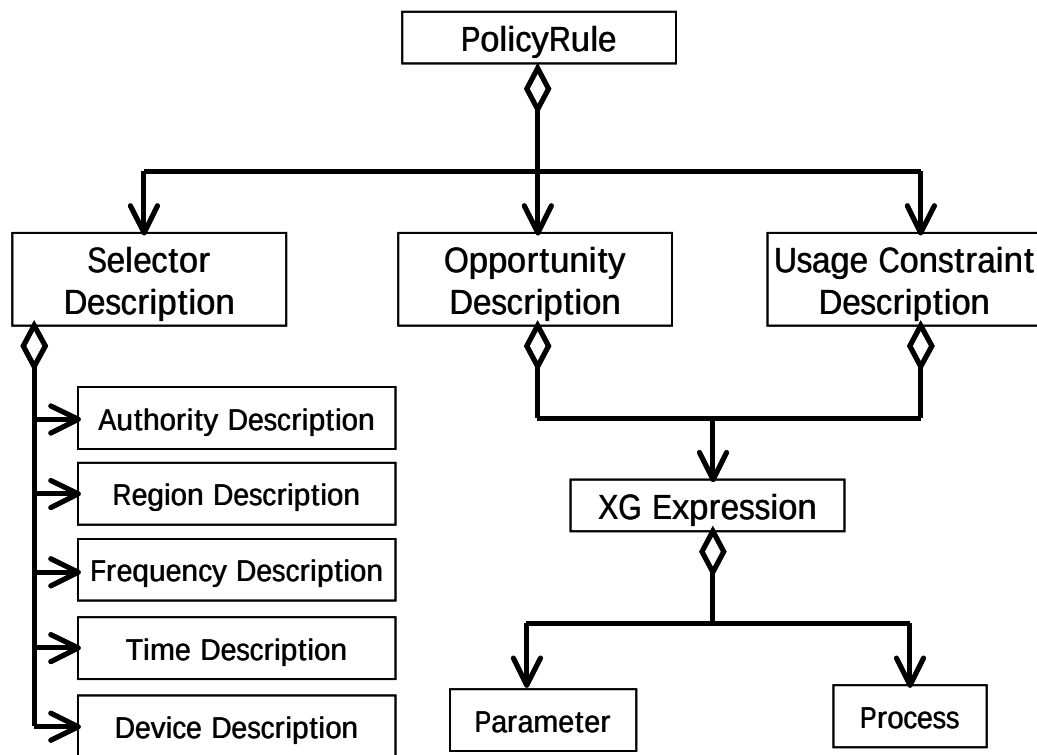


Figure 6: Structure of Policy Facts

The second fact in a policy rule is an opportunity description. This fact provides an expression that is used to evaluate whether or not a given environment and device state represents an opportunity for this policy rule. For example, it can express opportunities such as: “peak received power is less than -80dBm” or “if a beacon is heard at 823MHz.” The opportunity description is not evaluated unless the selector description from its policy rule matches. If a potential opportunity, called an opportunity instance, matches the opportunity description, a flag in the policy rule indicates if this is a valid or invalid opportunity. A valid opportunity indicates transmission that conforms to the usage constraint description is permitted. An invalid opportunity indicates that transmission is not permitted.

The final fact in a policy rule is a usage constraint description. This fact provides an expression that constrains the radio behavior, such as the emissions permitted and the corresponding sensing requirements when using the opportunity described in the policy rule. For example, it can express a usage such as: “transmit with a maximum EIRP of 10mW” or “maximum continuous on time must be 1 second and the minimum off time must be 100ms”

3.2.2 Extensible Semantics of Policy Parameters and Processes

As new radio platforms and policy sets are developed, they will have capabilities that were not envisioned when the policy language was developed. However, we will need to represent facts about these radios to fully describe selector, opportunity, and usage constraint descriptions that constitute policy. The policy language supports two generic constructs, namely, parameters and processes to represent policy concepts.

Parameter facts define all the values that are in the policy. They include values such as frequency bands, power levels, times and geographic areas. Parameters may be bound if the value is known, or unbound if the value is not yet known, such as a value that the radio platform is expected to bind.

Some policy rules will require the radio to perform certain functions to provide the information necessary to match an opportunity or usage constraint description. These are described using *process facts*. The process fact describes inputs and output parameters for the function (analogous to a function prototype in a programming language such as C) and possibly expressions constraining the relationship between the inputs and outputs. The radio is responsible for providing an implementation of the process. If a radio doesn't support a needed process, then it cannot use the opportunity that the policy rule describes.

In addition to these generic constructs, we have organized some of the key concepts in this domain into *ontologies* to provide a usable foundation. These ontologies include a structure of authority and delegation, frequency classifications, geopolitical regions and spatial descriptions, time, device capability descriptions, environment and device state descriptions, physical quantities, and units.

For example, an authority is an entity that has jurisdiction over some frequency, region, time, and set of devices and is authorized to create policy for that jurisdiction. An authority may be, for example, a regulatory agency or a primary user who is authorized to lease their spectrum to other users. A fact defines the authority and states its jurisdiction.

Some policy rules may apply only to a radio with a specific set of capabilities, either because the policy rule is designed for a specific type of radio or because it requires a particular set of parameter types or processes for the policy rule to be evaluated (e.g., supports geo-location, implements database access function, or maintains a history of parameter values). Such information is captured in a device description fact.

3.2.3 Meta-Policies

In absence of any other information, the set unions of the usage constraint descriptions from all the policy rules that represent valid opportunities apply. However, meta-policy facts may state relationships between policy rules that modify this logic. We have included three types of meta-policy facts: grouping, precedence, and disjunction.

Grouping simply creates a named set of policies so they can be referenced as a group. The group may be described either by explicitly listing the member policy rules or by creating an expression that describes the policy rules that are members of the group.

Precedence provides information on how to interpret conflicting policy rules. Two policy rules that have matching selector descriptions may also have opportunity descriptions that match a particular

opportunity instance. If both these opportunities represent a valid opportunity the usage constraints for both policy rules apply. However, if the policy rules disagree on whether the opportunity instance is valid, there is a conflict and the rule with the higher precedence is applied. Precedence may be defined between two policy rules or two policy groups. If one policy group has higher precedence than another, then all its member policy rules have a higher precedence than the member rules of the other group.

Finally, two policy groups may be disjunctive. If policy groups are disjunctive, then the policy rules in one or more of those otherwise applicable groups can be selected for application.

3.2.4 Policy Processing Logic

Policy processing, within the policy conformance reasoner for example, requires a set of rules describing how to process and interpret the policy facts. These rules govern the selection of policy rules that match a given selector instance; the selection of a policy rule that represents a valid opportunity given a selector instance and an opportunity instance; and the conformance of a given selector, opportunity, and usage constraint instance to the policy set.

The processing rules allow regulators to create policy rules that build on existing rules. Unless modified by a meta-policy, the union of all usage constraints from policy rules that represent valid opportunities is required for transmission to occur in that opportunity. Therefore, policy administrators do not have to enumerate policies for each band but may, instead, create broad policies and refine them where necessary.

For example, if a policy administrator wanted to constrain emission power in the television broadcast bands to -10dBm where an opportunity was detected, but additionally restrict emissions to have a 1MHz bandwidth in the frequencies³ represented by channels 58-63, only two policy rules need to be written. The first rule would set the -10dBm constraint on all television channels and the second rule would set the bandwidth constraint on channels 58-63. If a radio wanted to transmit on channel 61, it would have to conform to both emission constraints, as both policy rules apply. At a later stage, regulators may easily add a third policy rule that constrains new radios of a particular type from transmitting on channel 59 in selected regions, without modifying the other two rules.

We believe that the union of simpler rules (logical implications) scales well and offers flexible management of policies. In contrast, the use of procedural structures such as if-then-else conditions and while-loops can result in brittle policy structures over time as policy evolves. For example, a small change in policy can result in a large number of changes that must be reflected across several if-then-else branches or require the entire policy to be rewritten. Furthermore, an if-then-else structure can artificially constrain the order in which conditions are tested by a particular radio.

3.3 Policy Language Representation Requirements

A standard representation for the XG policy language is necessary so that regulators can encode policies in one language and all XG radios understand the encoded policy. This section discusses the features required for representing XGPL, provides an overview for some representations considered and introduces the OWL Web Ontology Language, the standard representation used for XGPL.

Spectrum policies have a complex structure with many dimensions and layers of exceptions that are difficult even for human interpretation. In selecting a language, one must ensure that the language is capable of capturing and potentially simplifying a number of aspects of this complex structure, including:

³ Supporting information such as the mapping of channel numbers to frequencies must exist elsewhere in the policy.

- *Inheritance*. Spectrum policy is very large and complex. The property of inheritance helps manage this complexity by enabling policy rules and properties to be extended and reduce the need for re-enumeration. For instance, rules for the 2.4GHz unlicensed band can inherit and extend rules for general unlicensed use.
- *Reification (rules about rules)*. Policy rules may make statements about other policy rules. For example, we can make a policy rule governing when or where a set of policies will apply.
- *Inference (derivable rules)*. There are rules that may not be explicitly stated, but follow from two or more rules. For instance, given that TV channels are 6 MHz wide, and that a policy exists to permit XG devices to transmit within a locally unallocated TV channel, one may infer that the maximum bandwidth for an XG device using an unallocated TV channel is 6 MHz. More common languages such as XML and IDL, while perhaps more straightforward for humans, do not facilitate such inference.
- *Extensibility*. It is imperative that the policy language be extensible in its vocabulary, structure, and semantics so that the language can adapt to express new types of policies as spectrum policy requirements change. Additionally, different countries may be concerned with different types of policies or different signal parameters, and it is crucial that the language be extensible to include their requirements.
- *Maintainability*. Policy will evolve over time. Small changes in policy should require only small, localized changes in the policy encoding and should not have ripple effects throughout the policy. Additionally, the language should make it easy to select policies that apply to a particular situation (i.e. time, location, frequency bands, device type). This argues for a declarative approach based on facts and rules instead of a procedurally based language as exceptions. We discuss this more below.
- *Scalability*. The policy language must scale to a wide range of devices that may have very different levels of resources available. The policy specification must be amenable to processing on devices with small memory and computational resources.
- *Standards*. Using a standard language representation is preferred as it enables reuse of libraries, applications, and tools previously developed. These tools reduce the cost of developing and using the language and increase interoperability between different tools. It is also expected that using a standard representation will facilitate the international acceptance of XGPL.

The language should be a declarative language based on facts and rules instead of a procedural language. One reason is maintainability as mentioned above. In a procedural language, policy would become a nested set of exceptions—many layers of *if-then-else* clauses. This creates two problems. First a small change in the policy may then affect many exception clauses, making it difficult to update policy. Second, the order in which exceptions are defined necessarily optimizes the policy for particular parameters. If a radio is concerned with other parameters, it will have a less optimal search.

Additionally, regulatory policy does not tell the radio what to do; it only defines what constitutes authorized use of the spectrum. However, enforcing a policy may require the results of a function that may be implemented in the XG radio. Using a declarative language, the policy can describe the function and specify rules based on its inputs and outputs without specifying the particular implementation of the function.

3.3.1 OWL

The World Wide Web Consortium's Web Ontology Language is a machine-understandable, semantic markup language. It is an extension of the Extensible Markup Language (XML) and the Resource Description Framework (RDF). OWL is based on DAML+OIL, a combination of efforts in the US (DAML—the DARPA Agent Markup Language) and the European Union (OIL - the Ontology Inference Layer) to create a machine-understandable semantic markup language. OWL provides a rich language for representing an ontology⁴, that is knowledge about the interrelationship of objects, in a manner that allows a machine to make inferences.

There are three variations of OWL. OWL DL provides the maximum expressiveness of OWL while providing well-understood complexity characteristics of description logics. OWL Lite is a subset of OWL DL that trades off expressiveness for processing complexity. For example, it restricts some of the constraints available in OWL DL, such as limiting cardinality restrictions to be 0 or 1, or the restriction of property constraints locally to a class. OWL Full supports the same language constructs as OWL DL, but relaxes some of OWL DL's restrictions that provide its computational guarantees. XGPL will use OWL DL as the computational guarantees are required and it needs more expressiveness than OWL Lite provides.

3.3.2 Why OWL for XG?

OWL provides a crucial set of features that match the requirements for representing the complex structure of spectrum policy. Specifically, OWL is extensible, and it supports reification, inference, and several object-oriented features such as multiple inheritance.

Powerful tools are being developed to process OWL that build upon earlier research in the knowledge representation, logic, and theorem proving communities in order to support the semantic web.

We chose to focus on markup languages for several reasons. First, the automatic conversion from a highly structured markup format to other non-markup formats is relatively easy; the reverse transformation is typically more difficult and non-uniform. Markup languages and tools to process them are widely adopted around the world and are continually evolving, as is evidenced by the Web. These Web markup standards satisfy the need for a representation that is suitable for cross-platform information exchange and processing across nations and organizations.

In an earlier work, Mitola and Maguire investigated a wide variety of languages to represent policy for cognitive radios. They concluded, as we do, that a knowledge representation approach is the most suitable for radio policy. They recommended the development of a new language based on the Knowledge Query and Manipulation Language (KQML), a language that has limited exposure outside the knowledge-based systems community. Since the time of their study, the World Wide Web Consortium has developed the OWL Web Ontology Language. Using a language that is a W3 Consortium Recommendation offers potentially wider adoption.

OWL's predecessor, DAML, has also been demonstrated to be an effective technology in other complex policy domains. The KAoS Domain and Policy Services project has successfully used DAML to represent distributed logistics policies for the DARPA UltraLog program.

In summary, OWL provides the features required to implement a machine-understandable semantic model. This enables the building of generic applications that depend on knowledge of the semantics of the content, including generic theorem provers and reasoning engines that enable deductive

⁴ An “ontology” is the knowledge about the relationships between objects. However, OWL also uses the term “ontology” to refer to an OWL file that encodes ontological information.

inference. OWL has the ability to represent the syntax of spectrum policy and the complex knowledge embedded in it. Additionally, OWL provides the ability to deduce policy constraints that are not explicitly stated for each specific case, but rather inferred from more general statements. Finally, we note that OWL is a Recommendation of the World Wide Web Consortium. Thus, we believe that OWL is an excellent long-term solution for the XG policy language.

Therefore, we provide an OWL representation of the XG policy language ontologies. We further show how to use these ontologies to specify machine-understandable spectrum policy separately from the implementation of the policy within particular radios.

3.4 Policy Ontologies

A number of ontologies that represent knowledge related to XG policy have been developed and represented using OWL. These ontologies have been made publicly available through a website and are also distributed along with the policy conformance reasoner software [PCR]. The concepts in these ontologies provide the foundation for expressing machine understandable policies, and additional concepts can be added to these ontologies to enhance the range of policies that can be supported.

We provide a summary of the ontologies here. Additional details on the content of these ontologies can be found in the XG Policy Language RFC [XGPLF].

- **Policy Language:** The “xgpl.owl” ontology specifies the central concepts for the XG policy language including classes and properties to represent policy specifications and how to group them. Policy specifications have a selector (specifying authority, frequency, region, time, and device class the policy pertains to), an opportunity description (an expression specifying the condition for an opportunity to exist), and a usage constraint description (an expression specifying constraints on the use of the opportunity).
- **Rules and Expressions:** The “xgpl-rl.owl” ontology describes an expression language for specifying opportunity descriptions and usage constraint descriptions, and a rule language to express policy processing logic. OWL semantics does not natively include rules and expressions, and the World Wide Web Consortium (W3C) is currently developing additional standards such as the Semantic Web Rule Language to address this issue. Future work should attempt to harmonize this ontology with such standards after they become available.
- **Parameters:** The “xgpl-param.owl” ontology specifies the Parameter class; device parameters governed by policy and other environmental parameters specified by policy must be individuals of this class.
- **Processes:** The “xgpl-proc.owl” ontology specifies the Process class; all functions and relations that are implemented by the device and used in policy specifications must be individuals of this class. Future work may attempt to harmonize this ontology with W3C the OWL Services ontologies being developed by the W3C.
- **Frequency:** The “xgpl-freq.owl” ontology specifies concepts related to frequency.
- **Geography and Geometry:** The “xgpl-regn.owl” ontology specifies concepts related to geographical and geometric regions.
- **Time:** The “xgpl-time.owl” ontology specifies concepts related to time.
- **Device Types and Capabilities:** The “xgpl-devc.owl” ontology specifies concepts related to device capabilities and device types.

- **Authorities and Delegation:** The “xgpl-auth.owl” ontology specifies concepts related to the structure of authority, their jurisdictions, and delegation relationships between authorities.
- **Physical Quantities:** The “xgpl-physq.owl” ontology is a utility ontology that specifies concepts related to physical quantities that are used in policy specifications.
- **Units and Conversions:** The “xgpl-physq.owl” ontology is a utility ontology that specifies concepts related to units and conversions.
- **Abstract Behaviors:** The “xgpl-abs.owl” ontology specifies concepts related to abstract behaviors (in terms of Parameter and Process individuals) that XG devices must support.
- **Environment:** The “xgpl-env.owl” ontology is a placeholder for specifying concepts related to the operational environment.
- **System-specific Extensions:** The “xgpl-sysect.owl” ontology is a placeholder for specifying concepts related to system-specific extensions.

These ontologies, currently populated with more than a hundred concepts, are able to support the encoding of a small range of notional XG policies. The above ontologies, represented using OWL, can be readily extended. Additional concepts can be imported from other standard ontologies as they become available. These ontologies must be developed further and populated with a richer set of concepts, based on inputs from domain experts – policy administrators, spectrum planners, and XG device manufacturers – in order to support a wider range of policies required for eventual deployment and maintenance.

3.5 Policy Conformance Reasoner (PCR) Software and Policy Utilities

A key component of the XG Policy Language Framework is the Policy Conformance Reasoner (PCR), which ensures that spectrum use proposed by the XG device is authorized by policy. BBN has developed a reference implementation of the PCR based on version 1.0 of the XG Policy Language Framework RFC. The source code for the PCR software and related utilities have been delivered to the participants of the XG program, and have also been publicly released with Government permission under a non-restrictive license.

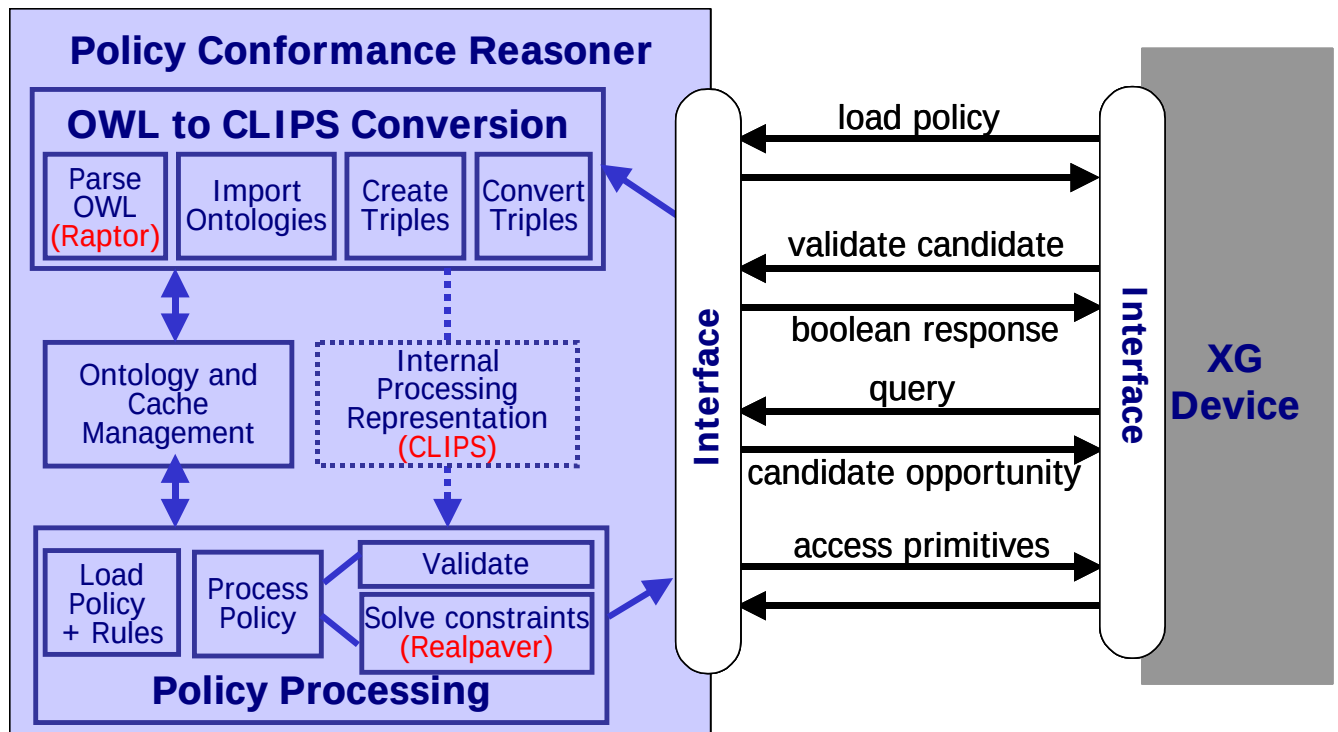


Figure 7: Policy Conformance Reasoner Software Architecture

The PCR performs the following three functions:

1. *Loads* (fetches and interprets) policy content encoded in the XG policy language that is available locally or across the network. Both the OWL representation and the shorthand representation introduced in the XG Policy Language Framework RFC are supported.
2. *Validates* the use of spectrum opportunities proposed by the XG device against the loaded policy content. The request must take the form of a fully bound opportunity, that is, all parameters that are required to validate the opportunity must be bound to values.
3. *Searches* for spectrum opportunities based on device queries that contain partially bound opportunities, and optionally device-specific constraints. Currently the search capability has been implemented as a standalone research prototype, which is to be integrated with the rest of the PCR software in follow-on work.

The PCR is based on a knowledge-based systems approach.

Loading policy content essentially populates a knowledge base with fact assertions and inference rules. Policy content may depend on background knowledge (other fact assertions and inference rules) that can be imported from existing ontologies. Many, but not all, of these facts express constraints on various parameters governing spectrum use by a radio.

Validation of proposed spectrum use involves making assertions based on what the device intends to do, applying the inference rules, and checking if the device's assertions are consistent with (i.e. do not contradict) the facts asserted by the policy. More generally, one may view this as an automated theorem-proving procedure.

Searching the knowledge base for potential opportunities that are authorized by policy involves making additional assertions on parameters that will satisfy the constraints specified by the policy. In general, one may view this as a constraint solving procedure. Several approaches are possible depending on the nature of the constraints (e.g. logical, integer arithmetic, real, linear, non-linear, or

a combination), performance requirements, and the desired solution quality. Typically no single approach works uniformly well across the range of constraints; therefore implementations must be customized to the requirements.

In order to validate and search for opportunities, the PCR requires information about the capabilities, current configuration and state of the device and its environment. For this purpose the device must provide an interface through which the PCR can access device primitives related to capabilities, configuration, and state information.

The software architecture of the PCR is illustrated in Figure 7. The PCR is implemented using the C/C++ language, and the current implementation depends on three freely available software packages (also implemented in C/C++):

1. Raptor, OWL/RDF parser libraries, which use the libxml2 libraries to parse XML and to fetch content over the Internet
2. CLIPS, a rule-based inference engine
3. Realpaver, a package for solving a range of non-linear real arithmetic constraints

The PCR interacts with the rest of the XG Device through a clearly demarcated interface, which is currently based on text strings exchanged across BSD sockets. Alternative interfaces can be created by directly accessing the lower level functions implemented by the PCR. For convenience, the interface is provided over two different sockets: one for policy management functions such as loading, and another to serve validation and search requests.

When the PCR receives a request for loading policy, it fetches and parses the policy using the Raptor libraries. Supporting ontologies that are imported by the policy and rules that encode the policy processing logic are also fetched and parsed. Sets of triples (RDF statements each containing a Subject, a Predicate, and an Object) are generated as a result of this processing. These triples are then loaded into CLIPS, and OWL processing (e.g. type inference) is done subsequently using rules we have implemented in CLIPS. Additional rules then convert the policy represented in OWL into an internal representation (quite similar to the shorthand notation) that is suitable for processing validation requests.

When the PCR receives a validation request, it asserts additional facts into CLIPS and runs the policy processing rules. If the assertions satisfy the loaded policy (i.e. there are no contradictions), then the request is validated. If the request fails to validate, diagnostic messages of conflicting facts in the policy can be output optionally.

When the PCR receives a search request, the request must include relevant selector and opportunity parameters. When making a query, the device can optionally include additional constraints of its own (e.g. opportunities with a bandwidth of at least 1 MHz) in addition to those imposed by the policy. The constraint expressions from all selected policy rules are then extracted, and output (along with any device constraints) in a form suitable for input to a constraint solver.

We have prototyped the search capability using the Realpaver software which can solve a set of non-linear real arithmetic constraints. Realpaver produces output in the form of a paving, a collection of at most k (a user-specified number) INNER and OUTER boxes, the union of which approximate all possible solutions that satisfy the constraints. INNER boxes represent valid opportunities usable by XG systems as is, whereas OUTER boxes must be processed further to extract valid opportunities. Alternative approaches to search and the integration of this capability with the PCR software are the subject of future work.

The PCR software can run either in the foreground or as a daemon in the background. The policy language ontologies can be fetched directly from their respective URLs, or the PCR can be

configured to use locally cached copies. A test harness called the PCR Client is included in the distribution along with an encoded policy example and sample validation requests.

In addition, format conversion utilities that enable the use of a shorthand notation (described in [XGPLF]) for representing policy have also been developed. General-purpose graphical editing environments for OWL are in their infancy. Until they mature and can be customized for editing XG policy, our conversion utilities offer a convenient text-based alternative to encoding XG policies.

Details on how to use the software and the related utilities, and a syntax specification for the PCR interface can be found in the distribution.

3.6 Encoded Policy Examples and PCR Tests

In this section, we provide an overview of the capabilities of the Policy Conformance Reasoner software through a sampling of the variety of policies it can handle, and a characterization of its performance for validation.

3.6.1 Opportunity Validation and Search using an Example Policy

The first example we describe simple policies that provide traditional opportunities to the radio without requiring sensing. In this example, Policy set E1 sets up a maximum power spectral density profile over a range of frequencies between 1.433 and 1.445 GHz. Additional constraints on the maximum continuous ON time and the minimum OFF time (between consecutive ON times) are also specified. This policy is illustrated in the top left corner of Figure 8.

We encoded this policy using the XG policy language and loaded it in the PCR. Using test scripts we emulated the behavior of an XG device by making opportunity validation requests to the PCR. The requests generated by the test scripts sampled the parameter space uniformly. The results of the validation test are illustrated in the top right corner of Figure 8 with green marks for valid opportunities and red marks for invalid opportunities. The PSD vs. Frequency plot indicates that the PCR is interpreting Policy E1 as expected.

In order to demonstrate the ability to incrementally modify policy, we add Policy E2, which modifies Policy E1 by adding a notch-out band between 1.442 and 1.443 GHz where no emissions are allowed by XG devices. The combined policy is illustrated in the bottom left corner of Figure 8. We encoded Policy E2, loaded E1 and E2 combined into the PCR and ran validation tests. The results plotted in the bottom right of Figure 8 show the presence of the notch-out band as expected.

Next we output the constraints set by the policies E1 and E2 in a form suitable for input to the realpaver constraint solver. The solver inputs and output for our example are shown in Table 1.

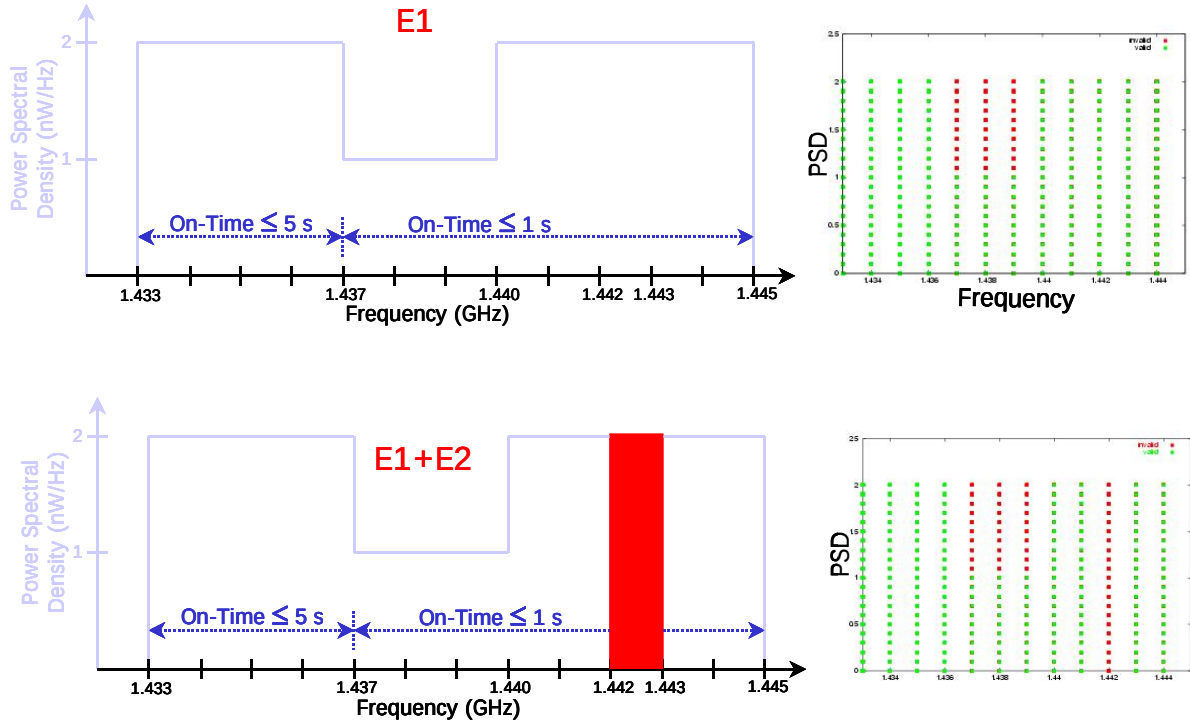


Figure 8: Policy E1 with PSD Constraints Modified by E2 to introduce a Notch-out Band

Table 1: Opportunity Search Based on Non-Linear Real Constraint Solving

Solver Input	Solver Output
<pre> Bisection choice = mn, mode = paving; /* realpaver tool directives */ Constants f0 = 1.433E9, f1 = 1.436E9, f2 = 1.440E9, f3 = 1.442E9, f4 = 1.443E9, f5 = 1.445E9; Variables XmitMin in [0.0, +oo[, XmitMax in [0.0, +oo[, PSD in [0.0, +oo[, OnTime in [0.0, +oo[; Constraints XmitMax >= XmitMin, /* background info */ XmitMin >= f0, /* selector */ XmitMax <= f5, /* selector */ /* conjunctions and disjunctions encoded using min and max */ min(f4 - XmitMin, XmitMax - f3) <= 0.0, /* notch-out */ min(PSD - 1.0, min(max(max(1.433E9 - XmitMin, XmitMax - 1.436E9), PSD - 2.0), max(max(1.440E9 - XmitMin, XmitMax - 1.445E9), PSD - 2.0))) <= 0.0, min(OnTime - 1.0, max(max(1.433E9 - XmitMin, XmitMax - 1.436E9), OnTime - 5.0)) <= 0.0; </pre>	Query: (requestOpp) Response: INNER BOX 1 XmitMin in [1437000000,1438111111.111111] XmitMax in [1438666666.666667, 1440333333.333333] PSD in [0, 1] OnTime in [0, 1] Precision: 1.67e+06, elapsed time: 10 ms

The solver returns 1024 boxes (by default) that contain all the solutions. The very first box is usable, but does not represent the best available opportunity for the radio. An issue for this approach is that when the problem is under-constrained, some solutions may be trivial or unusable by the device, and useful solutions contained within OUTER boxes require further narrowing. Although all solutions are contained within the 1024 boxes, subsets of combinations of the boxes are valid, but it is not practical to compute and test them all.

A more useful approach is to pass constraints from the device to the solver through the opportunity query. For example, in the above case, if we add a constraint that the device requires a 9MHz opportunity, a unique solution is returned.

In the XG Policy Language, usage constraints are included in the UseDesc class and these constraints can be extracted automatically. Therefore the solver can be integrated with the PCR by writing some glue software.

3.6.2 Other examples

We will briefly describe four other sets of policy examples that we have encoded and tested the PCR for its ability to load these policies and validate opportunities. In all these cases, the PCR is able to load the policies and the results of the validation indicate the policy is interpreted correctly.

3.6.2.1 *Sensing-based Policies in Adjacent and Overlapping Frequency bands*

The purpose of this example was two-fold. First, spectrum opportunities of interest to XG will involve the sensing of primary signals, and policies that enable such sharing are required. Second, it is desirable to be able to combine policies in frequency bands that are either adjoining or overlapping.

This example contains three sets: P1, P2, and P3. P1 applies to the frequency range 3.6-3.7 GHz, P2 applies to the adjoining frequency range 3.5-3.6GHz, and P3 applies to the frequency range 3.45-3.55 GHz, which overlaps P2.

P1 authorizes XG devices to transmit with a maximum PSD of 1nW/Hz if the incumbent signal power is sensed to be -105dBm. The corresponding values for P2 are 2nW/Hz and -80dBm. The corresponding values for P3 are 1.5nW/Hz and -95dBm.

All three policies impose additional constraints on the minimum look-through interval for sensing, maximum ON time, minimum OFF time, and maximum power leakage outside the band.

3.6.2.2 *Geographical Policies*

The purpose of this example was to specify policies for location-aware devices. Two policies P4 and P5 were specified that applied to the same frequency range but with different operational constraints within different regions (specified by latitude and longitude ranges).

3.6.2.3 *Time-based Policies – Opportunity Expiration*

The purpose of this example was to specify time-based policies. The ability to expire policies enables experimental policies that can then be replaced by other policies at a later time. Two policies T1 and T2 that applied to the same frequency and region were specified. T1 expired at midnight on a particular day and T2 (which relaxed transmission power constraints) took effect after T1 expired.

3.6.2.4 *Reuse of Television Spectrum*

The purpose of this example was three-fold. First, there is interest within the industry and the regulatory community to investigate the reuse of television spectrum, and therefore it is useful to consider policy examples for this scenario. Second, the ability to specify policies that involve a functional relationship between parameters is of interest. Third, it is useful to be able to specify policies with different constraints based on varying device capabilities.

This example included two policies TV1 and TV2 that applied to television channels 60-69. TV1 applied to devices based on simple energy sensing, and authorized a maximum PSD of -53 dBm/Hz if the sensed power was below -100 dBm. TV2 applied to more capable devices that employ sub-noise detection of DTV signals, and authorized a transmit PSD of $(-160 - \text{sensed power})$ dBm/Hz provided the maximum sensed power of the DTV signal was below -107dBm, and the maximum transmit PSD did not exceed -40 dBm/Hz.

3.6.3 Policy Tools Capabilities Summary

The capabilities of the policy tools developed in the XAP project are summarized in Table 2. With further development, we believe that the technology can be transitioned to XG systems in follow-on work.

Table 2: Policy Tools Capabilities Summary

Capability	Status
Adjacent band policies	Yes, can express, validate, and have example
Overlapping policies	Yes, can express, validate, and have example
Simple geographic constraints	Yes, can express, validate, and have example
Simple temporal constraints, expiration	Yes, can express, validate, and have example
Device capability based policies	Yes, can express, validate, and have example
Off-line merge of overlapping policies	Yes, can validate, and have example
Extensible, declarative, incrementally modifiable	Yes, based on semantic web and rule engine
Validation performance and scalability	Yes, sub-second response on PC with 10s of policies and 100s of facts, significant gains possible from straightforward implementation changes
Disjunction and other meta-policies	No, can express, but not validate, and complicates search (selectors are a practical workaround)
Candidate crosses policy region (different set of rules apply to different parts of same candidate)	No, but a practical workaround exists (use selectors, e.g. UWB specific selector), not a show-stopper
Supporting background ontologies	Yes, but quite sparse, and needs enhancement driven by further policy use cases, proposed for Phase 3
On-line merge/revoke of policies	No, but proposed for Phase 3
Policy set inheritance	No, but possible through language extension
Authority delegation and user management structure	No, but possible through language extension
Validation engine	Yes, software already available, more performance and robustness enhancements, and CORBA interface planned for Phase 3
Search engine	No, working prototype code, anticipate integration early in Phase 3
Syntax and logical consistency checker	Yes, offline based on OWL consistency checkers
Human-friendly policy encoding tools	No, have engineer-usable CLIPS-based surface notation along with tools to convert to/from OWL, but a graphical development environment friendly to spectrum-manager proposed for Phase 3
Domain-specific policy interaction analyzer	No, do not anticipate need for Phase 3, but needed eventually
Trusted policy distribution and management system	No, do not anticipate need for Phase 3, but needed eventually, may leverage trusted download for software-defined radio

3.6.4 Validation Performance

In this section we report a preliminary evaluation of the current prototype PCR software. A purpose of the evaluation was to serve as a rough guideline for current capabilities while developing the XG policy technology roadmap described in Section 7.

The results of loading various combinations of policy sets that we described earlier are shown in Figure 9. The combinations considered are

- Sensing-based policies P1 (Section 3.6.2.1)
- Sensing-based policies in adjoining and overlapping bands P1+P2+P3 (Section 3.6.2.1)
- Geographic policies P4+P5 (Section 3.6.2.2)
- Time-based policies T1+T2 (Section 3.6.2.3)

- Maximum PSD and Notch-out policies E1+E2 (Section 3.6.1)
- Re-use of television spectrum policy TV1 (Section 3.6.2.4)
- All of the above

Metrics that we have plotted in Figure 9 include the number of common processing rules (tens), the number of policies, the number of precedence meta-policies after inference (tens), the number of selected policies for the opportunity to be validated, the total number of facts (tens), the time for preprocessing the encoded policy to the internal form, and the time for validate an opportunity. The experiments were preformed on a 800MHz Pentium III laptop with 256MB of RAM.

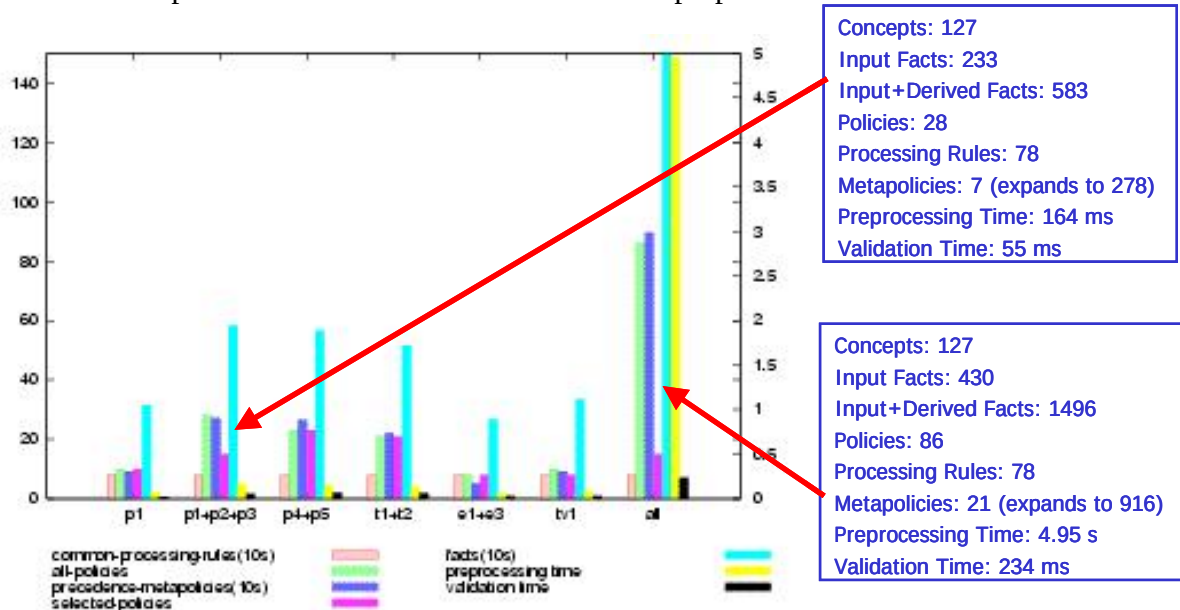


Figure 9: Validation Performance of the Policy Conformance Reasoner

The key observations we made were the following:

- PCR prototype is able to handle over a hundred concepts and over a thousand facts, covering an interesting range of policies of interest to XG, and provide a sub-second validation time response on with modest computing hardware requirements
- The current prototype loads and processes the policies upon each request. By caching the results of preprocessing the policies (e.g. processing precedence meta-policies has a quadratic complexity in the number of policies), significant performance gains can be achieved.
- It is beneficial to hierarchically organize policies so that we can limit the number of policies that are selected at any time. This will reduce the number of facts over which the inference rules must be run for each validation.

Other software enhancements that can be considered in future work are documented in the TODO list included in the PCR software distribution.

4 Abstract Behaviors

In order to facilitate the verification, validation, and accreditation process for a wide range of diverse XG systems, it is desirable to identify and formally specify the minimal set of behaviors that a system must implement to be able to safely share or use available spectrum and conform to the requirements of regulators and spectrum assignees. In short, if a device meets this minimal set of requirements, its opportunistic use of the spectrum will be “safe” from the perspective of regulators and incumbent spectrum assignees.

The challenge in specifying a minimal set of behaviors is, of course, that the range of envisioned XG systems is large. An XG system could be a simple transceiver capable of employing IEEE 802.11 signaling on any one of sixteen different frequencies, or an XG system could be a fully software-programmable radio capable of transmitting at any frequency in the range from 3KHz to 300 GHz using any one of a hundred encoding techniques. The challenge is how to specify behaviors, without requiring extraneous functionality for the simple system and without under-specifying the behavior of the fully programmable system. At the same time, the set of behaviors should be all of those necessary and sufficient for regulators (and incumbent spectrum assignees) to deem a radio safe and compliant, so that a new set of guidelines is not required for certifying each new XG implementation.

The solution to this challenge is to specify abstractions – *behaviors, interfaces and information objects* – that allow considerable flexibility in how the abstractions are implemented. An abstract behavior is a specification of the behavior of an XG system. Abstract behaviors interact with each other and with the system via abstract interfaces. An abstract interface is an interface that specifies a set of functions (but not their implementations), and an abstract information object specifies the information that passes across the functional interface.

In this document we specify a set of abstract behaviors that XG-compliant systems must implement; we include these behaviors in the **accreditable kernel** and argue that it is those behaviors that are necessary and sufficient to satisfy policy requirements and demands of spectrum assignees without requiring the development of a new set of guidelines for certifying each new XG implementation.

Related to the notion of an creditable kernel is the notion of **traceability**. An XG system implementation, therefore, must support a one-to-one mapping between the external physical behavior (e.g. emission profile) of the system and the instances of particular abstract behaviors that the system implements. Linking external behavior (which is fundamentally what spectrum regulators and incumbent assignees care about) with internal abstract behavior provides both a standard for minimalism (if it doesn’t affect external behavior, it isn’t a function or behavior in the minimal set) and suggests an approach to auditing radio behavior, to ensure the radio is well behaved.

Following this rationale, we have developed a set of guidelines to help determine whether a given abstract behavior must be included in the creditable kernel or not. In order to be included, the behavior must satisfy at least one of the following:

- It is required to support opportunistic sharing of spectrum
- It is required to support policy-defined operation
- It provides traceability from policy through behavior to emissions

The abstract behaviors are only intended to provide guidelines for the high-level system design of the individual XG systems. Although the abstract behaviors will cover the important attributes that characterize XG systems, they are not intended to be detailed design- or implementation-level specifications of particular XG systems.

In the following subsection, we present an UML model of an abstract XG radio system. We present three kinds of abstractions - Interfaces, Behaviors, and Information Objects - in terms of which an abstract system can be described. We provide a high level object-oriented description of the abstract XG radio system in terms of these three kinds of abstractions. For a more detailed treatment, please refer to the draft Abstract Behaviors RFC that not only describes each of the three kinds of abstractions in significant detail, but also describes a reference radio design based on the abstractions.

4.1 An Abstract Model of the XG Radio System

In this section, we present an abstract model of an XG radio system (radio plus software on it and its interactions with the outside world). That may appear to be at odds with our goal in this document of defining a minimal set of essential behaviors, our *accreditable kernel*. The abstract behaviors that reside within the creditable kernel, however, interact (through XG Interfaces) with other portions of the XG radio system. So in defining the creditable kernel, we also need to present how it interacts with the portions of the radio outside the creditable kernel. That's the focus of this section.

Since our goal is to define the creditable kernel, and leave the details of the rest of the radio completely open to innovation (and indeed, to keep the creditable kernel open to innovation too), this abstract model is a careful combination of concreteness and vagueness. In general, we seek to specifically describe what functionalities have to be present, yet we do not specify the details of how those functionalities are realized.

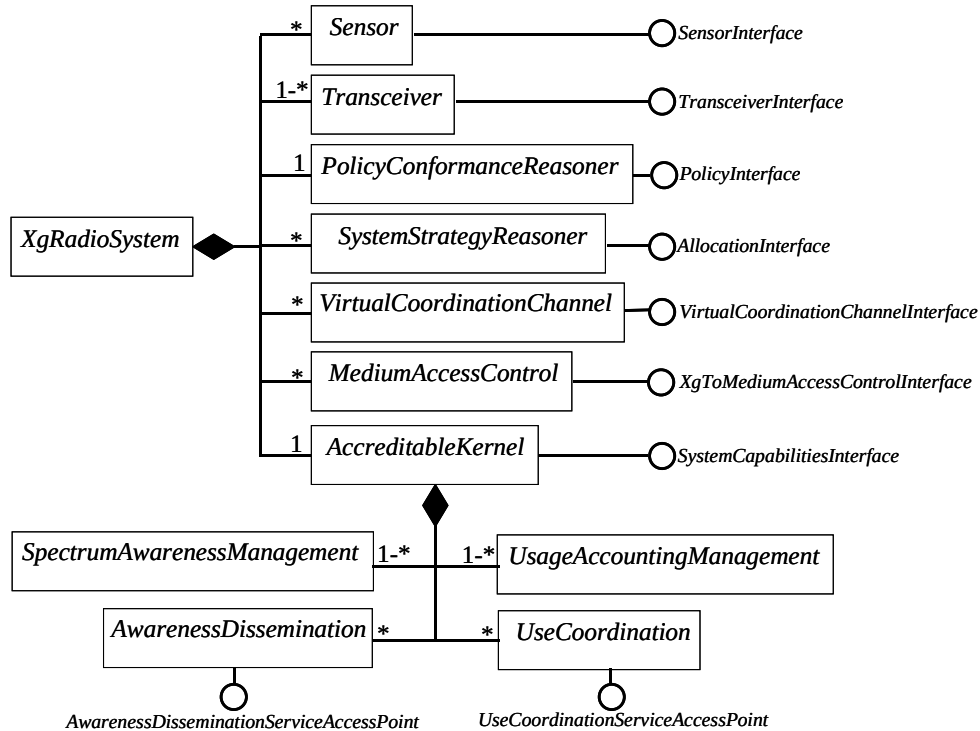


Figure 10: XG Radio System Abstract Model

In this section, we define a UML class for a radio, entitled XgRadioSystem. This class is illustrated in Figure 10 and is composed of all the components that an XG radio might need. Many components are optional, but some components must be present, and some components may be present one or more times.

The purpose of this representation is to illustrate and represent the basic hardware and primitives of a host radio system that enables opportunistic use of spectrum (e.g., RF front end, DSP hardware, system

software including the OS, middleware, and libraries, and primitives for networking protocols, waveform agility, and beam forming). This configuration of components is only one of many ways that one might choose to modularize the functions of an XG radio.

In general, each component of the XgRadioSystem class has a related interface. (It is important to keep in mind, however, that other classes may also implement a particular interface. The mapping shown in Figure 10 simply indicates that the class, if present, implements the associated interfaces). The accreditable kernel interacts with the various components through these interfaces, using information objects. So in this section we describe what each component does, and define a generic class, XG Interface, from which each component's interface will be derived. In anticipation of defining the behaviors of the accreditable kernel, we define a generic class of behaviors, XG Behavior. Finally, we define three generic classes of XG Information Objects from which all information objects used in the interfaces will be derived.

4.1.1 Subsystems of the XG Radio System

The various components of the XgRadioSystem class serve different purposes. We describe each component and its function here.

4.1.1.1 *AccreditableKernel (required; one instance)*

The AccreditableKernel class has the following key functions:

- (i) Implement the included XG abstract behaviors and provide access to them;
- (ii) Manage access to the state, configuration, and capabilities of the radio; and
- (iii) Manage access to primitives implemented within the radio that can be governed by policy.

Every instance of the XgRadioSystem must have an instance of the AccreditableKernel.

The AccreditableKernel class implements an interface (the SystemCapabilitiesInterface) through which it provides access to primitives deemed necessary to ensure that control of the radio's visible behavior is located in the AccreditableKernel and that the behavior is controlled in a fashion consistent with XG policies. These primitives include Parameters and Processes described within the XG Policy Language Framework (see [XGPLF]).

The accreditable kernel also has an association with every other subsystem in the radio for each primitive implemented in that subsystem (these associations are not shown in the diagram above).

4.1.1.2 *Sensor (optional; may have more than one)*

The Sensor class provides situational information to the radio about the spectral environment at a given location and time. The sensor outputs need not be limited to the RF spectral environment; the provided information could include a variety of other values such as geo-location, temperature, and proximity to specific targets that could be used as parameters for system policy.

Sensors are optional because radios may learn through other means, such as configuration or through dissemination protocols, what spectrum use local operating rules authorize. Alternatively, the sensor functionality may be implemented within the transceiver subsystem. There may be more than one sensor because a radio may sense different information using different devices.

The Sensor class implements the interface called SensorInterface.

4.1.1.3 *Transceiver (required; at least one)*

The Transceiver class provides radio communications capabilities, including the RF front-end and the baseband signal processing functions. In general, we expect these communications capabilities to be agile and that the interface to the Transceiver will reflect this agility.

Instances of XgRadioSystem contain at least one Transceiver, by which we mean it must have at least a receiver or a transmitter capability. (Without a transceiver, we don't have a radio!).

The Transceiver implements the interface called TransceiverInterface. In cases where the transceiver is also a sensor, it may make sense for the transceiver to implement SensorInterface as well.

4.1.1.4 *PolicyConformanceReasoner (required; one instance)*

The PolicyConformanceReasoner class determines whether proposed spectrum use is consistent with accredited policy (e.g. approved by a relevant regulatory authority and incumbent spectrum assignee), knowledge of the local spectrum (e.g. information from the Sensor class), and other background knowledge. Note that the PolicyConformanceReasoner primarily determines if proposed use is acceptable – the task of developing proposed usages is left to other subsystems, for example, the SystemStrategyReasoner. The PolicyConformanceReasoner may, however, perform additional policy functions. It can support queries on policy to extract authorized usages for a given situation, support filtering policy information to obtain a subset of policies that apply to a current situation, generate machine proofs that a given usage conforms to policy, and manage the loading and revocation of policy sets.

Instances of XgRadioSystem must contain a PolicyConformanceReasoner, as the PolicyConformanceReasoner implements the functions that determine if the XG radio is in compliance with any policy restrictions and the abstract kernel depends on conformance reasoner to accredit usages

The PolicyConformanceReasoner class implements the interface called PolicyInterface.

4.1.1.5 *SystemStrategyReasoner (optional; may have more than one)*

The SystemStrategyReasoner class determines the system's strategy for opportunistic spectrum sharing given the constraints of sensor information, regulatory and system policy constraints. This reasoner is aware of system-specific optimizations and tradeoffs and has control over the radio platform.

In many ways, the SystemStrategyReasoner is the complement of the PolicyConformanceReasoner. The conformance reasoner determines whether a particular type of spectrum use is authorized by policy in the current environment; the SystemStrategyReasoner determines what opportunities to use spectrum exist in the current environment that are suitable for the XG radio. It is important to note that the strategy reasoner is not the only place in an XG radio that can identify or allocate opportunities: opportunities can be found from other nodes through an awareness dissemination protocol or this function can be incorporated within a medium access control protocol.

The SystemStrategyReasoner class, therefore, is one of several classes that can perform the opportunity allocation function, and so it implements the interface called the AllocationInterface. This class can access the state, capabilities, and configuration information through the SystemCapabilitiesInterface.

4.1.1.6 *VirtualCoordinationChannel (optional; may have more than one)*

In many situations, XG radios need to communicate with neighboring XG radios to coordinate awareness of the spectrum (e.g. are a set of frequencies unused at both sender and potential receivers?) and coordinate use (e.g., jointly determine which frequency bands will be used). Coordination channels can be pre-configured, discovered, or created when needed. Furthermore, the channel may be

implemented using waveform-level signaling, as a specialized MAC layer, or even at the application layer using higher layer protocols using out-of-band network access.

The VirtualCoordinationChannel class represents the logical communication channel used for this purpose.

The VirtualCoordinationChannel class implements the interface called VirtualCoordinationChannelInterface.

4.1.1.7 MediumAccessControl (optional; may have more than one)

The MediumAccessControl class includes higher layer functionality such as neighbor discovery, topology management, and link scheduling and sends/receives Layer 2 protocol data units to the transceiver. In an opportunistic spectrum-sharing environment, the higher layers need to interact with the XG radio. For example, links must be scheduled such that communicating peers select and use common opportunities that are deconflicted from opportunities used by other nodes.

The MediumAccessControl class implements the interface called XgToMediumAccessControlInterface.

4.2 XG Interfaces

In this subsection, we sketch the functions of generic Interface class and briefly mention the purpose of each of the major interfaces in XgRadioSystem.

As illustrated in Figure 11, all XG interfaces extend the Interface abstract class, or its subclass ServiceAccessPoint. In addition to the seven interfaces associated with the subsystems described in Section 4.1.1, we define two additional interfaces that are associated with the protocol behaviors included in the AccreditableKernel.

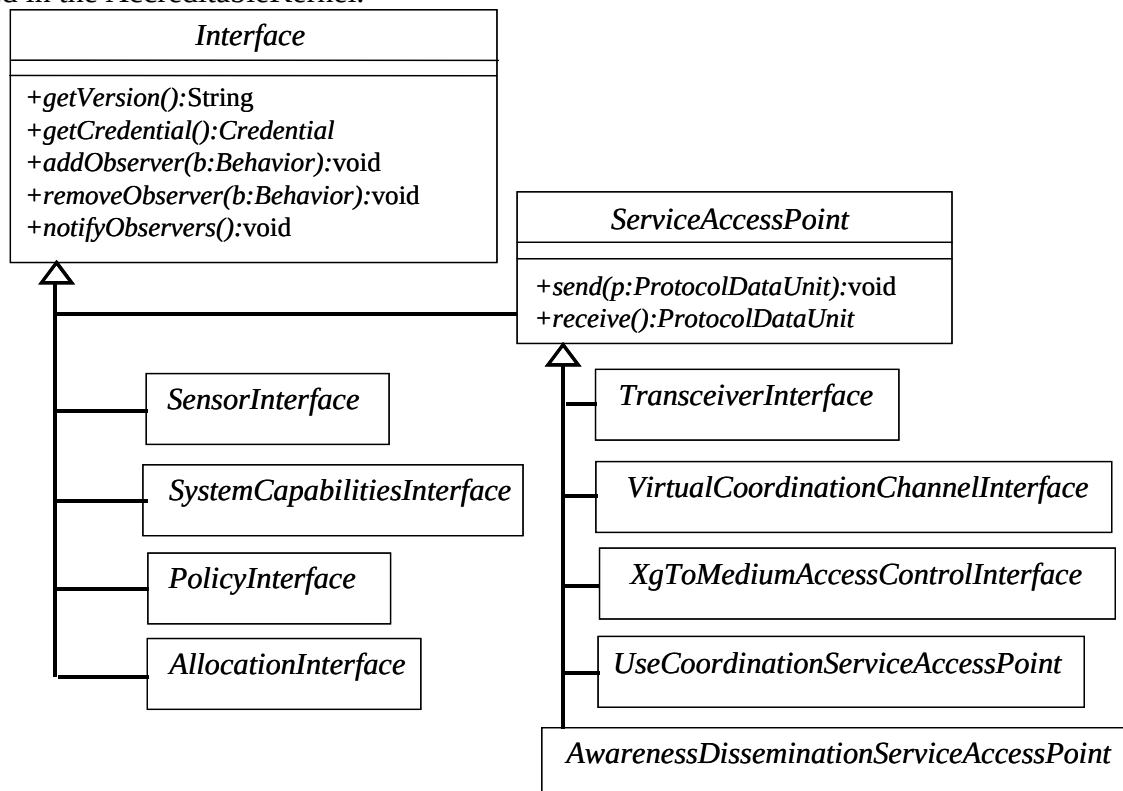


Figure 11: XG Interfaces

The Interface class provides five methods:

- *getVersion*, a method used to learn the version of the Interface; and
- *getCredential*, a method used to get credentials of the class that implements the interface, for example, digital certificates that assert that the implementation has been accredited. *getCredential* allows trust establishment mechanisms to be established to work across XG interfaces. This enables various components of the radio to be accredited and enhanced separately, and the use of particular combinations to be governed by policy if needed.
- a set of three methods that behaviors use to register and unregister at an interface and to be notified of events. Behaviors and Interfaces together implement an Observer–Subject design pattern [DP]. A behavior can register or unregister itself at an interface by calling the *addObserver* and *removeObserver* methods respectively. The *notifyObservers* method is called, which in turn calls the *update* methods on all instances of Behavior that are registered with the interface. For example, the SpectrumAwarenessManagement behavior can register with the SensorInterface and the AwarenessDisseminationServiceAccessPoint in order to know when new sensed awareness information or protocol based awareness information is received.

The ServiceAccessPoint class extends the Interface class with two additional methods:

- *send*, a method to send a protocol data unit. Note that the definition of protocol data unit (PDU) is somewhat broader in this document than is traditional – in this context, a PDU is protocol data plus a set of associated attributes which often includes information about what channel the PDU is to be sent on
- *receive*, a method to receive a protocol data unit, typically invoked by an observer in response to a notification

XG abstract behaviors interact with each other and with the system through the following extensible interface set:

1. **SensorInterface:** The interface through which sensed awareness (of the operational environment, i.e. information such as location and spectrum) is accessed and sensing behavior is controlled.
2. **TransceiverInterface:** The interface through which the agile XG transceiver (i.e. parameters such as transmit power, frequency, waveform, and beamform) is controlled and emission constraints are conveyed.
3. **SystemCapabilitiesInterface:** The interface through which the capabilities, the current configuration, and state of the XG system can be accessed.
4. **PolicyInterface:** The interface through which opportunity instances are validated against applicable regulatory, incumbent spectrum assignee, and system policy. This interface may additionally provide access to policy information and policy management directives.
5. **AllocationInterface:** The interface through which a specific opportunity is allocated for use from all available opportunities.
6. **VirtualCoordinationChannelInterface:** The interface through which virtual control channels (which carry XG protocol data) are managed and accessed.
7. **XgToMediumAccessControlInterface:** The interface through which the MAC layer can interact with the XG abstract behaviors (for example, to support link setup and maintenance, contention management, and framing).
8. **AwarenessDisseminationServiceAccessPoint:** The interface through which the AwarenessDissemination protocol behavior can be accessed. The information accessed through this interface is processed spectrum awareness that is acquired from or is to be disseminated to other nodes by the protocol (rather than the individual protocol data units exchanged by a protocol implementing this behavior over virtual coordination channels)

9. **UseCoordinationServiceAccessPoint:** The interface through which the UseCoordination protocol behavior can be accessed. The information accessed through this interface include higher level protocol directives resulting in acquisition and release of opportunities (rather than the individual protocol data units exchanged by a protocol implementing this behavior over virtual coordination channels)

XG systems that implement these interfaces are anticipated to inherit and extend these interfaces and the related classes in order to support real-time operation, access control, error or exception handling, enhanced features and system-specific optimizations.

4.3 XG Behaviors

XG systems implement the abstract behaviors necessary to enable opportunistic use of spectrum, policy-defined operation, and traceability. As a guideline to identifying which abstract behaviors are necessary, we use the following rationale. XG systems must minimally agree on how they characterize spectrum conditions (spectral awareness) and on the opportunities available to use spectrum (opportunity instances), and must support behaviors that enable the dissemination of spectral awareness and opportunity information. In addition, for traceability, they must account for emissions they make in terms of the policies that authorize each emission, and the valid opportunities that were identified to enable the emission. Finally, they must support behaviors that allow systems to coordinate the use of opportunities either to form a data communications channel (at Layers 2 and above), or to avoid opportunities selected by other XG systems.

XG abstract behaviors extend the Behavior abstract class. As illustrated in Figure 12, the subclasses of Behavior include:

- InternalBehavior, the class of behaviors that are *internal* to the XG system
- ProtocolBehavior, the class of behaviors that involve communications with agents external to an XG system. The protocol behaviors use virtual coordination channels to exchange XG protocol information, and access to the protocol behaviors can be made through a ServiceAccessPoint interface.

Behaviors interact with other behaviors and the rest of the system through Interfaces. Behaviors and Interfaces implement the Observer-Subject design pattern. A behavior can register itself at an interface, and the behavior implements a generic *update* method that can be called by the interface to notify the behavior of any state changes.

We have identified the following four XG abstract behaviors:

1. **SpectrumAwarenessManagement:** an InternalBehavior, which describes how opportunity information is acquired, identified, represented and disseminated within and across XG systems. This behavior encompasses awareness information gained from sensing, configuration, and through AwarenessDissemination instances.
2. **AwarenessDissemination:** a ProtocolBehavior, which can be used by XG systems to share opportunity awareness information.
3. **UsageAccountingManagement:** an InternalBehavior, which enables emissions to be traced to a valid opportunity enabled by policy. This behavior is responsible for ensuring that every opportunity is validated for policy conformance prior to its use. This behavior is also responsible for ensuring that all parameters governing operation are correctly bound to the values specified within the validated opportunity.
4. **UseCoordination:** a ProtocolBehavior, which allows XG systems to coordinate the use of selected opportunities with other (XG and non-XG) systems.

A natural question to ask is why these four behaviors in particular are included in the accreditable kernel and why some others (notably, allocation) not included. Recall from Section 1 that abstract behaviors must satisfy at least one of the following to be included in the accreditable kernel:

- It is required to support opportunistic sharing of spectrum
- It is required to support policy-defined operation
- It provides traceability from policy through behavior to emissions

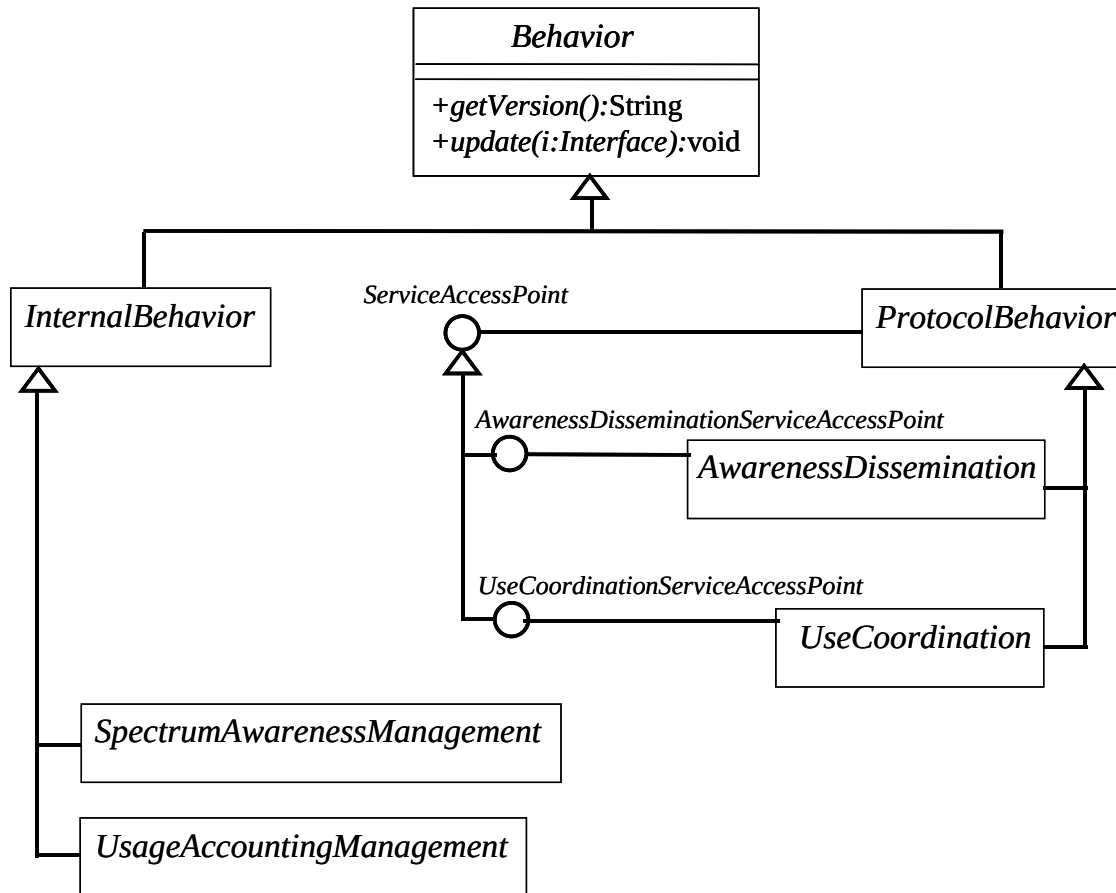


Figure 12: XG Internal and Protocol Behavior Classes

A radio without awareness of opportunities will cease to be an XG radio (i.e. it cannot perform opportunistic spectrum sharing); therefore, SpectrumAwarenessManagement is included within the accreditable kernel.

UsageAccountingManagement addresses the policy-defined operation and traceability requirements directly, and is included within the accreditable kernel.

In addition, traceability requires that trustable dissemination and coordination protocol behaviors (though not necessarily particular protocols) may be required by policy in some circumstances. An example of this requirement can be found in proposed approaches for the reuse of public safety bands through the use of beacon signals that authorize use. These two protocol behaviors (one for trusted dissemination of what's available to share and another for trusted signaling of opportunity acquisition and release) are therefore included in the accreditable kernel because they relate to policy conformance. Clearly, these behaviors are required only in those XG radios that make use of particular opportunities enabled by policy due to the implementation of these protocols.

Identification and allocation of opportunities need not be a trusted function. A tool for finding opportunities which devises wonderful spectrum use most of the time and occasionally suggests something infeasible/illegal is perfectly reasonable, provided that the infeasible opportunities are never instantiated. It is the barrier to instantiation that is the accreditable kernel's function – not the exploration of opportunities. And leaving the exploration of opportunities outside the accreditable kernel allows for freer innovation.

4.4 XG Information Objects

We organize XG Information Objects into three abstract classes as illustrated in Figure 13: `PolicyDefinedJoinPoint`⁵, `Expression`, and `ProtocolDataUnit`.

The `PolicyDefinedJoinPoint`, (described further in the [XGAB]) is a generalization of two abstract classes: `Parameter` and `Process`. Instances of the `PolicyDefinedJoinPoint` class are primitives that are described using the XG Policy Language Framework (see [XGPLF]), and implemented within the XG radio system. The `AccreditableKernel` through the XG System Capabilities Interface provides access to these primitives.

Instances of the `Parameter` class are parameters that govern the operation of the XG radio system, and instances of the `Process` class are methods, procedures, or relations that are implemented by the XG radio system.

The `ParameterCollection` is an abstract (and arbitrary) collection of `Parameter` instances. Subclasses of `ParameterCollection` include `Credential`, `Awareness`, `UsageRequest`, and `OpportunityInstance`.

Instances of the `Credential` class encapsulate security and trust management information such as cryptographic keys and authentication information. `Credential` instances are exchanged across XG interfaces for trust management. The specification of the security and trust management architecture is not within the scope of this document.

The `Awareness` class encapsulates situational awareness information including spectral awareness, network awareness, and possibly temporal and geo-spatial information as well. Instances of the `Awareness` class are created and managed by the XG Spectrum Awareness Management behavior. Subclasses of the `Awareness` class include `SensedAwareness` that encapsulates awareness gained by the XG system through sensing, and `ProtocolBasedAwareness`, which encapsulates awareness acquired by the XG system through the use of awareness dissemination protocols.

The `UsageRequest` class and `OpportunityInstance` class respectively encapsulate information in the request for, and in the characterization of, spectrum use. `OpportunityInstanceCollection` is an abstract collection of instances of the `OpportunityInstance` class. Instances of `UsageRequest` can be sent (to the `SystemStrategyReasoner`, for example) over the `AllocationInterface`, and an `OpportunityInstanceCollection` that includes a list of opportunities may be returned. The radio system can choose opportunities from this collection, and then pass the selected instance of `OpportunityInstance` over the `PolicyInterface` to have it validated (i.e. checked whether it is authorized by applicable policy). A `ValidOpportunityInstance` object encapsulates validation status and associated credential information of an `OpportunityInstance` object that it contains. The XG Usage Accounting Management behavior ensures that operation (e.g. emission) conforms to the parameter values set within the `ValidOpportunityInstance` with the appropriate status and credentials returned by the `PolicyInterface`.

⁵ In computer science, a *join-point* is a point in the flow of a program. In Aspect Oriented Programming, a *pointcut* is a set of join-points. Whenever program execution reaches one of the join points defined in the *pointcut*, a piece of code (called advice) associated with the pointcut is executed. We envision a `PolicyDefinedJoinPoint` to be points within the XG radio where behavior governed by policy is accessible.

The Expression abstract class encapsulates predicate expressions that involve PolicyDefinedJoinPoint instances. The expression language can be specific to the particular XG implementation. Subclasses of the Expression class include the PolicyFilterSpecification class, which encapsulates expressions that specify a filter on policy instances based on arbitrary criteria of interest, and the PolicyConstraints class, which encapsulates expressions that specify constraints that apply to the XG radio system. These expressions can be, for example, used by the SystemStrategyReasoner to interact with the PolicyConformanceReasoner.

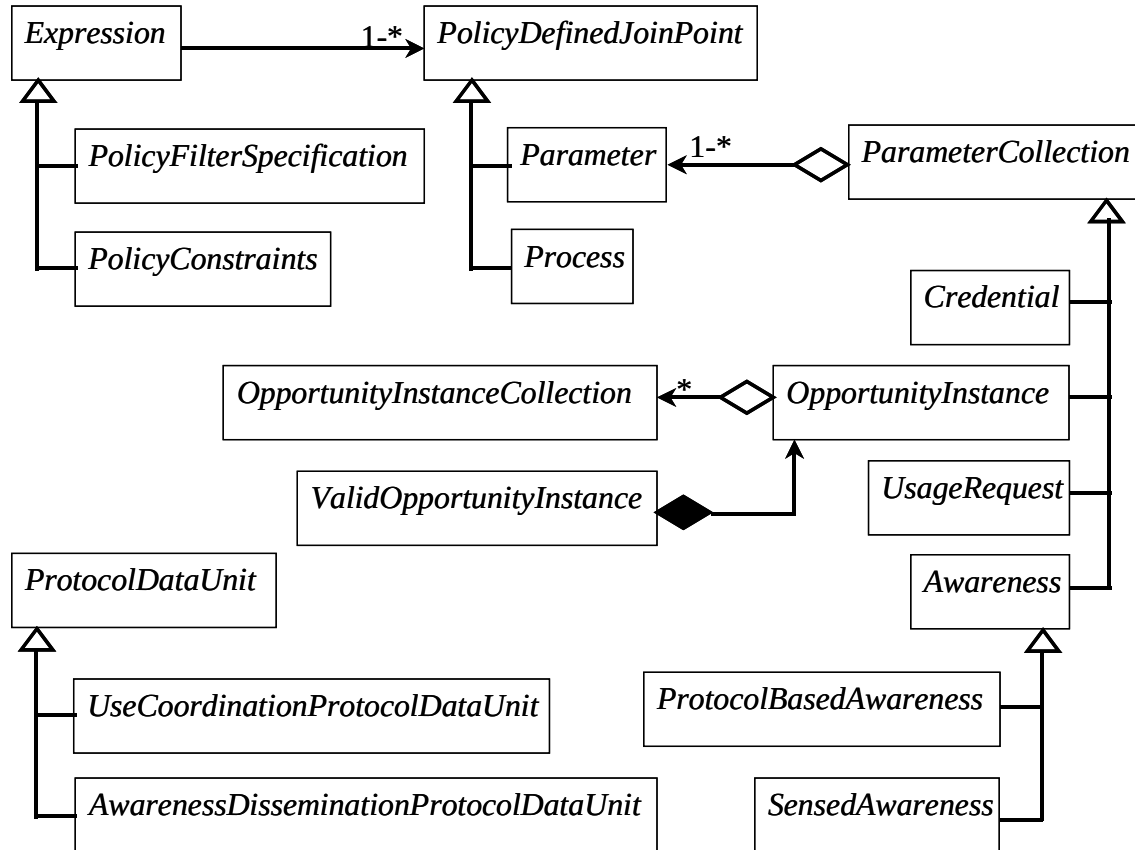


Figure 13: XG Information Objects and Their Associations

The ProtocolDataUnit is an abstract class that encapsulates information exchanged across ServiceAccessPoint interfaces (and any attributes of that information, required to effect an exchange). In particular, two subclasses of ProtocolDataUnit, namely, the AwarenessDisseminationProtocolDataUnit and UseCoordinationProtocolDataUnit are exchanged over the VirtualCoordinationChannelInterface by instances of the XG protocol behaviors (AwarenessDissemination and UseCoordination). The extension and implementation of these abstract classes are specific to the protocols used by the particular XG radio system.

4.5 XG System Interactions

There are three key system interactions that relate to opportunity awareness, allocation, and use.

The SpectrumAwarenessManagement behavior creates and manages Awareness objects, objects that provide information about the environment around the XG radio. To get the information necessary to create, update, modify, or delete Awareness objects, the behavior acquires SensedAwareness objects via the SensorInterface and ProtocolBasedAwareness objects from the AwarenessDisseminationServiceAccessPoint interface. The latter interface is implemented by the

AwarenessDissemination behavior, which exchanges spectrum information (abstractly modeled as AwarenessDisseminationProtocolDataUnit instances) with other systems via communications channels modeled by the VirtualCoordinationChannelInterface. These interactions are illustrated in Figure 14.

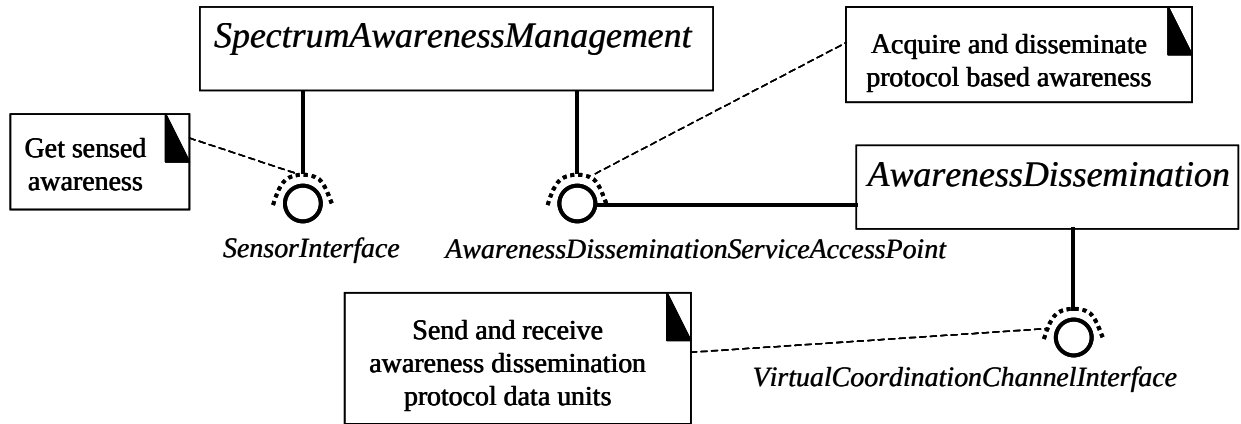


Figure 14: Spectrum awareness is gained through sensing and protocol behaviors

Once awareness of opportunities is acquired, the next step is to identify and allocate opportunities for use by the radio system under policy constraints. As illustrated in Figure 15, the SystemStrategyReasoner performs this function. The SystemStrategyReasoner can access policy information through the PolicyInterface, and it can access the capabilities, configuration, state, and primitives within the radio system through the SystemCapabilitiesInterface. In particular, this includes access to the XG SpectrumAwarenessManagement behavior.

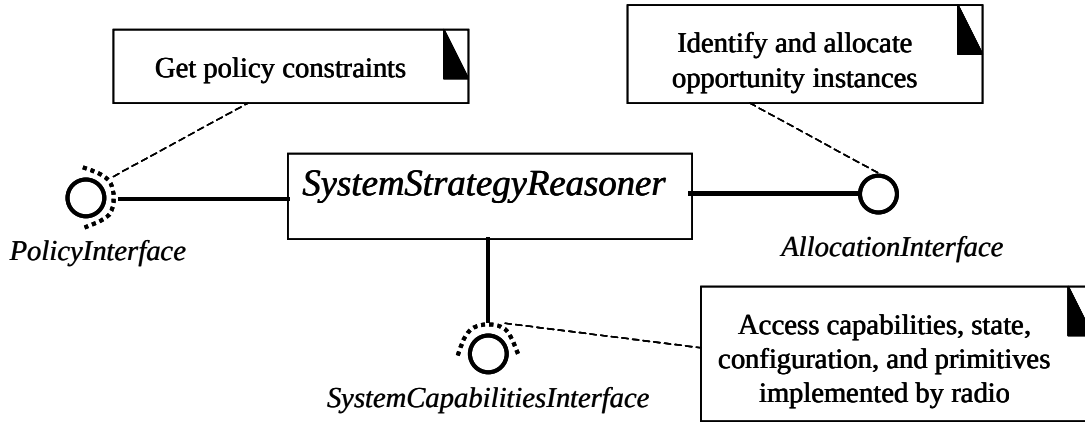


Figure 15: The System Strategy Reasoner identifies and allocates Opportunity Instances for the Radio; it may additionally access radio primitives to modify system behavior

Once opportunities have been identified and allocated based on awareness of opportunities, one or more can be selected and used in a manner that is authorized by policy. The UsageAccountingManagement behavior performs the function of ensuring that OpportunityInstance objects are presented to the PolicyInterface prior to use, and that only ValidOpportunityInstance objects returned by the PolicyInterface with proper validation status and credentials are actually used.

The UsageAccountingManagement behavior through the SystemCapabilitiesInterface asserts that subsystems actually use the parameter values contained within the ValidOpportunityInstance, before providing it to the TransceiverInterface. Furthermore, authorized use of spectrum may entail the employment of a protocol to coordinate with other nodes. For this purpose, the UsageAccountingManagement behavior interacts with the UseCoordination protocol behavior through the UseCoordinationServiceAccessPoint.

The interactions related to the use of opportunities are illustrated in Figure 16.

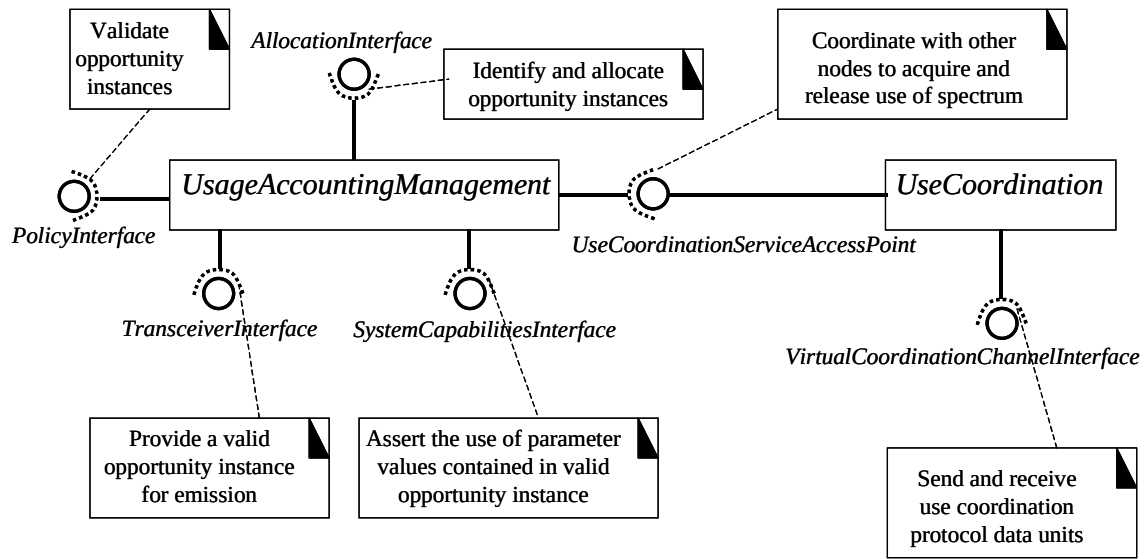


Figure 16: Opportunity instances are allocated, validated against policy, then used after asserting that all parameter values are set appropriately and after coordination with peers

A design sketch for a reference implementation of an XG system based on these abstractions is provided in the draft XG Abstract Behaviors RFC [XGAB].

5 Modeling and Simulation of Protocols for XG

During the XAP effort, we have refined and enhanced the XG Evaluation Platform (originally developed during the earlier XMAC effort). This software, developed in the OPNET environment, along with documentation, has been made available to all XG performers. The XG models and simulations developed by BBN in the context of this platform have also served the role of a proxy XG system for developing the XG framework, the abstract behaviors in particular.

5.1 XG Evaluation Platform (XEP)

XEP is an interacting set of modular mechanisms for opportunistic spectrum access. It consists of several new mechanisms working cohesively to implement a complete opportunistic spectrum access (OSA) system. In designing XEP, our objectives were:

- Backward compatibility. To develop a complete system solution, capable of exploiting opportunistic spectrum access even if the upper layers are not aware of this capability (e.g. using legacy MACs).
- Simplicity. The goal was a straightforward realization of opportunistic spectrum access that forced us to develop all the essential mechanisms and none of the optimizations. A simple system is a good benchmark against which to test optimizations, and provides a clear starting point.
- Modularity. We have sought to divide up the functionality such that adding new features or optimizations is easy, without destroying the general architecture.

Figure 17 shows the XEP system architecture. From the XEP perspective, the node is divided into three layers: physical, XEP Management, and MAC-and-above layers.

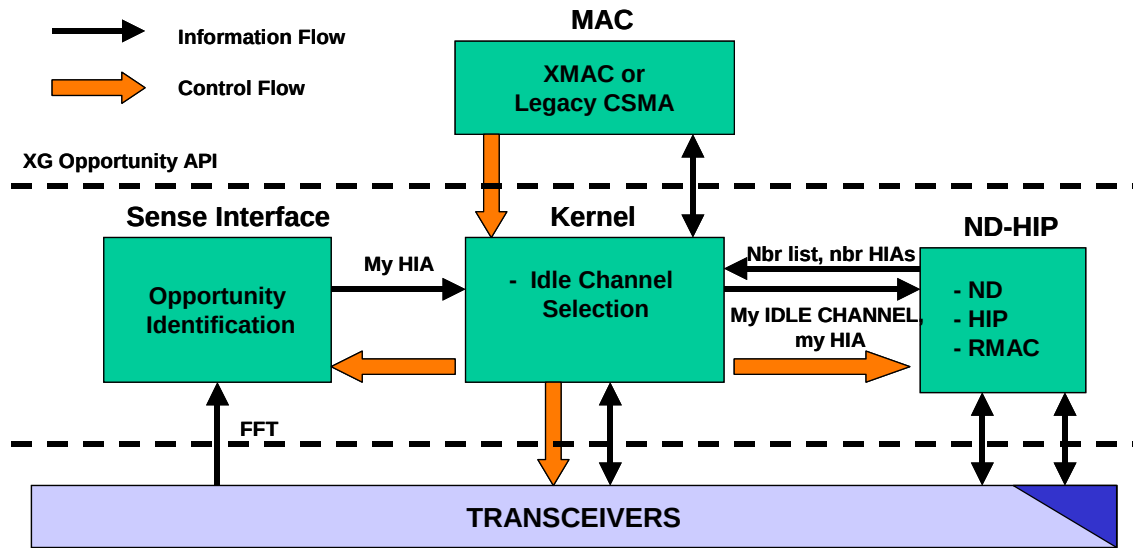


Figure 17: XG Evaluation Platform System Architecture

The physical layer consists of two transceivers. The first transceiver is frequency agile and can be tuned to different frequencies and waveforms by the XEP management layer. The second transceiver is tuned to the coordination channel. We assume that the coordination channel is allocated a priori and is known to all nodes. Each transceiver is capable of successfully decoding (receiving) one packet at a time. In addition, the physical layer includes a sensing device that periodically reports to the XEP management layer the primary nodes' energy over the frequency bands of interest.

Above the physical layer sits the XEP management layer. Among its functionalities are:

- Estimating the available frequencies in the node's neighborhood (opportunity identification).
- Discovering other XG nodes in the neighborhood and selecting those of them that are appropriate neighbors.
- Informing other XG nodes of this node's frequency observations.
- Selecting a default set of frequencies to become the node's IDLE channel. The IDLE channel is the frequency band(s) and waveform(s) that the node is tuned to when it is not transmitting, receiving, or preparing for reception of a pre-scheduled packet.
- Sharing this opportunistic node's IDLE channel identity with its neighbors so that neighbors know how to reach it.
- Serving as a front-end for the MAC layer. Beyond simply transmitting and receiving, the interface between XEP and the MAC layer allows the MAC layer to take an active role in channel selection and management.

The XEP management layer divides these functions across three modules: XEP kernel, Sensing Interface, and Neighbor Discovery and Hole Information Protocol (ND-HIP). The XEP kernel module centralizes this layer's decisions and presents a well-defined interface to the MAC layer. It also handles most physical layer functions. The Sensing Interface module is in charge of determining the presence of transmission opportunities. And the ND-HIP module is in charge of disseminating OSA information among opportunistic nodes.

In the experimental results in Section 5.3 we assumed the MAC layer was unaware of XG (e.g. a legacy MAC) and XEP was entirely responsible for finding spectrum to meet the MAC layer's needs. However, we expect that in many cases the MAC will be XG-aware and that the MAC will indicate to the XEP kernel the capacity the MAC needs.

Figure 17 also shows the information flow in XEP. The physical layer reports measured energy in various frequency bands. The Sense Interface module receives this information and converts it into a Hole Information Array (HIA). The HIA is a vector where each entry corresponds to a frequency band and indicates if the band can be opportunistically used. One should view the HIA as a joint product of the physical sensors and the Sense Interface, as the physical layer may have to be programmed by the Sense Interface to detect transmissions. The Sense Interface reports the HIA to the XEP kernel.

Concurrently, the NDHIP module exchanges HIA information with other opportunistic nodes and provides the HIA information from the node's neighbors (and perhaps peers farther away). Combining its local HIA information with that of its neighbors, the XEP kernel develops a map of how the spectrum is being used in its area, and selects the node's IDLE channel(s).

The XEP kernel provides the NDHIP module with both its HIA and IDLE channel information for the NDHIP to propagate it to other opportunistic nodes. This environment is described in greater detail in the software documentation [XEP].

5.2 Protocols for Opportunistic Spectrum Access

In this section, we describe a set of protocols for supporting spectrum agility in a Mobile Ad Hoc Network (MANET). These protocols were developed at BBN as part of the DARPA neXt Generation (XG) program, and have been implemented within an OPNET simulation model. Our main objective is to study the benefits of spectrum agility, not necessarily to design the most robust or efficient solution. We shall also briefly discuss some insights based on our experience using this simulation model.

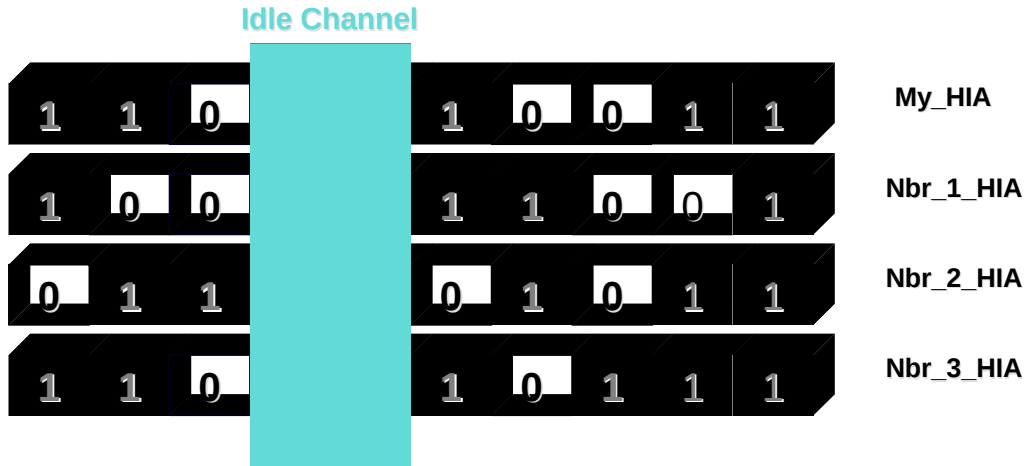


Figure 18: Idle Channel Selection from Hole Information Arrays

The target scenario consists of a set of primaries emit radio signals in a band, which is segmented into channels (we call them *frequency slots*), and a set of secondaries seek to self-organize into an ad hoc network using “holes” in the spectrum assigned to the primaries. As mentioned above, the problem has many variants and is very hard in general. Therefore, we shall make assumptions and design choices to simplify the problem. We assume that the primaries are chatty⁶, that XG secondaries have two transceivers, one of which is always tuned to a dedicated *coordination channel*, and that they have a mechanism that can tell a secondary node from a primary one. Finally, we use a contention-based protocol for medium access rather than a scheduled scheme (e.g. TDMA) for reasons of simplicity.

An XG secondary node senses the channel and maintains a time-varying *Hole Information Array (HIA)*. The HIA contains, for each channel in the band, an entry 0 or 1 indicating, respectively, whether the channel is free (a “hole”) or is occupied (a “wall”).

Each node periodically broadcasts its HIA as part of a control message called the *Hole Information Protocol (HIP)* packet. One purpose of the HIP packet is to serve as “beacons” or “Hello” messages for the purpose of discovering neighbors, as in many ad hoc routing protocols (e.g. OLSR). Specifically, a node considers another node as its neighbor if the moving window average of the number of received HIP packets exceeds a threshold. The HIP packets are sent on the dedicated *coordination channel* assumed allotted to the secondary nodes.

The HIP packet is also used for selecting the *idle channel*. Recall that this is the channel to which a node is tuned when it is not transmitting. Clearly, the idle channel for a node M must be chosen such that all 1-hop neighbors of M can transmit to M on it. That in turn implies that if *c* is an idle channel, then M and all of its 1-hop neighbors should have a hole in *c*. Thus, each node collects the HIA’s from its neighbors, and finds overlapping holes, and assigns the idle channel for itself. This is illustrated in Figure 18. As shown there, the idle channel need not be a single frequency slot. It can be a contiguous set of slots.

Once the idle channel is selected, it is included in the HIP packet so that all its neighbors know. Also included is the transmission power of the HIP packet so that neighbors can estimate the path loss (based on received power) and the waveform used to listen to packet preambles. The selection procedure is specified in more detail in [XEP].

⁶ We assume primary nodes are not silent for extended periods, and therefore, XG nodes can discover their presence.

We now describe briefly our MAC protocol. It is based on the well-known idea of Carrier Sense Multiple Access (CSMA), but the basic algorithm needs considerable modification in the presence of spectrum agility. A brief description is as follows. When a node S wants to send a data packet to node R , S first picks a set of frequency slots F that it perceives as holes (based on R 's HIP packet) for both R and S . It then tunes to R 's idle channel, say $I(R)$, and performs carrier sensing. If $I(R)$ is free for more than a certain amount of time, S sends a Ready-To-Send (RTS) on $I(R)$ and tunes to F and waits. R ensures that F does indeed reflect its holes and responds (if appropriate) with a Clear-To-Send (CTS) on F . R remains tuned to F for the DATA packet. Once S gets the CTS, it responds with the DATA packet and R responds with the ACK.

Unlike IEEE 802.11 MAC, the RTS/CTS is not effective for collision avoidance since each node may listen on a different idle channel. A node does not maintain a Node Allocation Vector (NAV) table based on the RTS/CTS packets overheard because it is not likely to hear such packet unless it is the intended recipient (RTS/CTS packets to nearby nodes will be transmitted over different frequency slots). Despite this, the RTS/CTS exchange is important for the selection of frequency slots and traffic flow coordination. However, better methods, perhaps based on TDMA are required going forward.

An issue that arises for large dense networks when the coordination channel is of low capacity is that the HIP packets may saturate it. To mitigate the problem, we have designed a lightweight protocol called *Rendezvous MAC (R-MAC)*. In R-MAC, a sender makes a prediction of the next time it will send a HIP packet and includes this information as part of the HIP packet. Neighbors avoid transmitting on the "reserved time". No carrier sensing, RTS/CTS or complex backoff procedures are needed. The simplicity is possible due to the largely periodic, and hence predictable nature of the HIP packets.

While the above solution sketch for spectrum agility does enable opportunistic spectrum access, it clearly leaves something to be desired. The need for an a priori dedicated coordination may not be realistic in many environments, and the question is if we can design a protocol that does not require it. It is also clear that CSMA has fundamental limitations with spectrum agile protocols and we need to investigate TDMA based algorithms, which is quite challenging. And if TDMA is used, there would be a need to disseminate hole information further than just 1 hop and this brings into question scalability issues.

5.3 Simulation Results

We used OPNET to simulate our XG system model and protocols running underneath a 'legacy' MAC system. The results are presented here.

We used a CSMA MAC to test that our system – while being simple – provides a complete solution that is able to work with any existing and future MAC implementing the opportunity API specified in [XEP]. For example, a legacy MAC that is sensitive to timing information tied to the packet's transmission rate will need to specify their desired transmission rate to XG. If this rate is achievable, XG will create IDLE channels with that rate.

For simplicity, we implemented an unreliable MAC (no ACK) when a node transmits a packet if the channel is free or backs off for a random time if the channel is busy. Nodes also backoff upon completing a packet transmission to prevent some nodes from capturing the channel. Even with this very simple MAC we were able to show a good performance enabled by using XG.

Table 3 summarizes the default simulation parameters for both primary and opportunistic nodes. Unless stated otherwise, our simulations consist of 60 opportunistic and 10 primary nodes placed randomly in a square area of 0.6 miles by 0.6 miles. The primary nodes are fixed and not always active. When they are active, they continuously transmit packets using the ON/OFF cycle shown, and record the period of time that they experienced interference by opportunistic nodes beyond the maximum tolerance. Their

transmission range is equal to 2 km, which is consistent, for example, with current cellular systems. A random mobility model is supported for the XG nodes; their speed can vary from 0 - 10 mph, with nodes bouncing back each time they reach the area boundary. The results we present in this section, however, are limited to static XG nodes. Their transmission range is 250m, which is consistent with current WLANs such as those based on 802.11. Opportunistic nodes generated a new packet to transmit as soon as the previous transmission is completed. This corresponds to a saturation case, particularly challenging for an unregulated CSMA MAC without feedback-based load balancing. Each packet's destination is chosen randomly among the one-hop neighbors. We collect statistics on the total number of packets successfully received by each node.

Thus, we collect two main figures of merit: the total aggregated one-hop throughput (sum of all bits received by each node), and the maximum interfered time (the time a primary node measured interference above its tolerance).

Table 3: Default simulation parameters

General Parameters		Primary Node Parameters		Opportunistic Node Parameters	
Name	Value	Name	Value	Name	Value
Simulation area	0.6 x 0.6 miles	Bandwidth	19MHz	Speed	0-10 mph
Simulation time	200 s	Transmit PSD	-26 dBm/Hz	Coordination Channel	200 kHz
Base Frequency	2.3 GHz	Avg. On time	160 ms	Max Transmit PSD	23 dBm
Total Bandwidth	100 MHz	Avg. OFF time	180 ms	Target PSD (policy)	-62dBm/Hz
Unassigned Bandwidth	5 MHz	Transmit range	2km	Transmit PSD	0.625 mW/MHz
Background noise	-174 dBm/Hz			Transmit range	250 m
				Target SNR	12 dB
				Sensing period	8 ms
				Sensing history	256 ms
				Sense threshold	-138 dBm/Hz

Finally, the physical layer assumes some capabilities such as sub-noise detection (enabled by a DSP intensive synchronization stage). Thus, if a large processing gain is used, the sensing range could be smaller than the transmission range. The propagation model assumes an attenuation model proportional to the fourth-power of the distance.

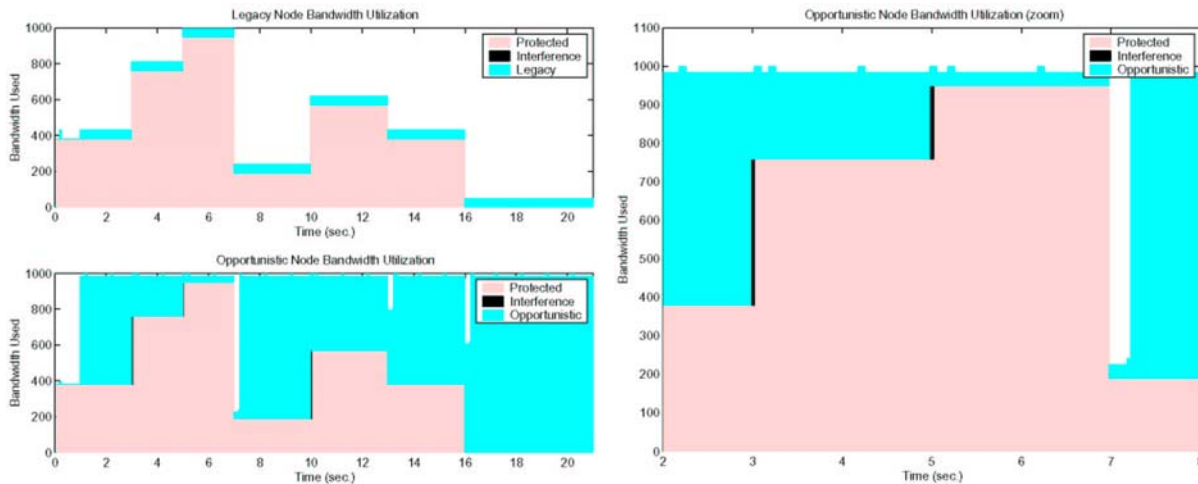


Figure 19: Spectrum utilization by Legacy (upper left) and XG (lower left) systems given the primary system utilization is 40%. A more detailed snapshot of the spectrum utilization by the XG system is shown on the right. Bandwidth is measured in terms of number of 100kHz channels used.

To show how our system exploits unused spectrum, experiments were conducted on a very simple network consisting of 4 nodes in a line, separated by 250m each. The primary nodes' tolerance was

decreased so that all the opportunistic nodes were inside the area where the primaries are protected by policy. Since we wanted to accentuate the negative impact of the coordination channel and modulation-type requirement, we increased the size of the coordination channel to 1.5MHz and the maximum transmit power of the opportunistic nodes to 40dBm.

Figure 19 shows how our algorithms take advantage of the unused portion of the spectrum. For these plots, we have divided the time into 40ms intervals and consider a frequency band used if a transmission occurred over any part of the 40 ms interval.

The upper left plot shows the way a legacy (i.e. current, fixed spectrum allocation) system – running the same MAC – used the frequency. The primary nodes are assigned 95 MHz of spectrum, so the legacy radios can use only the remaining 5 MHz. The frequency band has been divided in frequency slots of 100 KHz each, and the number of frequency slots occupied by each – and not the actual frequencies – is shown. The lower area represents the primary nodes utilization. For example, in the interval $\langle 0, 3 \rangle$ two primary nodes are active occupying 380 slots (i.e. 38 MHz). Similarly, in the interval $\langle 16, 21 \rangle$ no primary node is active. Averaging over the interval $\langle 1, 21 \rangle$ (i.e. after initialization) and over their assigned frequency bands, the primary nodes' present an utilization of 40%. Looking at the legacy node's bandwidth utilization, we can see the legacy nodes always use the same amount of bandwidth (the unassigned 5 MHz) and are constantly transmitting, with the exception of the interval $\langle 0, 1 \rangle$ (initialization) when they are only active for short bursts sending neighbor discovery beacons.

The lower left plot illustrates the way XG radios exploit the available spectrum. After the one-second-initialization period is completed, we see that XG fills the spectrum gaps almost completely, except for minute periods of time. Dark lines represent the periods where XG radios interfere with primary nodes. This interference is due to the latency on sensing the channel. It may take up to two sensing intervals (8 ms each) for the radio to detect that a primary node has become active, resulting on interference periods of up to 16 ms per each time a primary node becomes active. However, since our time step is 40 ms, we paint the whole 40 ms interval as 'interfered'.

The right plot is a 'zoom' of the lower left plot (XG spectrum utilization) for the interval $\langle 2, 8 \rangle$. This allows a more detailed examination of the XG system's behavior. First, we may notice that our XG system does not occupy the entire 1000 frequency slots all the time. Typically there will be a small gap, corresponding to the coordination channel's 15 slots. Only when HIP packets are sent (periodically every second and also event-driven upon primary nodes' activation at times 3s and 5s) XG will occupy the entire spectrum. Thus, the coordination channel and XG control packets reduces the DATA throughput achieved. Second, we see that upon a primary node's activation (for example at time 3 sec.), XG transmissions will interfere with primary nodes until their local sensors detect the primary node signal, and before any control packet is sent they will stop transmitting on the primary node's frequency band. This implies that they will not be able to communicate with any neighbor whose IDLE channel contains parts of that band (all the 4 nodes in our example). The XG nodes will then recompute their IDLE channel and include it together with their new HIA information in a HIP packet that is broadcasted after some random time (to avoid collisions from all opportunistic nodes sensing the primary node's signal at the same time).

Thus, in Figure 19 (right) we notice that the coordination channel is used after the interference period ends. So, from the time the primary node is detected until a HIP packet with new IDLE channel information is received, the nodes will not be able to reach neighbors (due to invalid IDLE channels).

The fact that we don't see white gaps in the spectrum occupancy at times 3 sec. and 5 sec. indicates that the nodes recover new information and resume transmission in less than 40 ms. Thus, the periods of communication disruption in XG are small. Third, when at time 7 sec. primary nodes become inactive, we see that there is a period (around 300 ms) when the system is unable to fully occupy the spectrum.

Locally detecting this opportunity is not enough to use it - we need the neighbor nodes to be able to transmit over it. Therefore, XG nodes must wait until receiving neighboring nodes' HIA updates (in this case once every second) declaring the opportunity before recomputing their IDLE channel. In the figure, around time 7.3 sec. we see the HIP packets being sent, and only after that the IDLE channels are recomputed, another round of HIP packets with the new IDLE channel information are sent, and the XG nodes start using the new opportunities. Note that during this transition there is no transmission interruption since XG nodes tune to the new IDLE channel only after the HIP packet transmission has been completed. In between (i.e. after waiting for a spreading time to avoid collision, for the coordination channel to become free, and for the transmission delay) the XG nodes are still tuned to their last-advertised IDLE channel. Overall, XG reacts fast to new opportunities. It could even react faster if HIP packets are sent immediately after detecting new opportunities, similar to the case of losing opportunities being used by someone's IDLE channel, but this is not advisable since it may lead to congestion on the coordination channel. Thus, only critical (i.e. causing communication disruption) events trigger extra HIP transmissions.

The two lower curves of Figure 20 plot the total throughput achieved by the legacy and XG systems for different levels of bandwidth utilization by the primary nodes. For example, the experiment shown in Figure 19 corresponds to the 40% utilization data points. Since we set a large coordination channel of 1.5 MHz, it is expected that when the primary nodes utilization approaches 100%, the legacy system (able to transmit over 5 MHz) outperforms the XG network, which can only use 3.5 MHz for data transmission. It can be seen that under a bandwidth utilization of 40%, XG gives an order of magnitude performance improvement over legacy systems, even when assuming a generous allocation bandwidth allocation for the latter (5 MHz of exclusive spectrum usage).

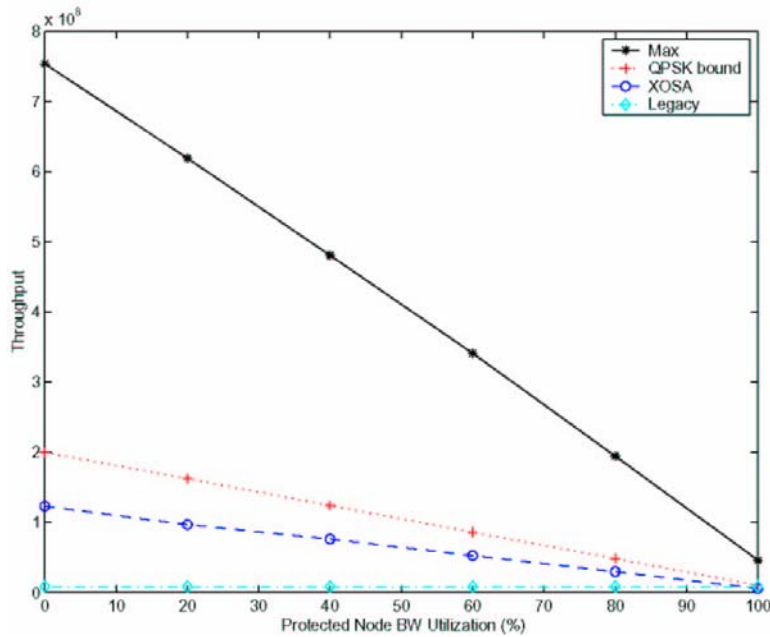


Figure 20: Comparing throughput achieved by XG and Legacy systems. The maximum achievable using a theoretical bound (based on Shannon capacity), and the maximum achievable if only QPSK modulation is allowed are also shown.

For comparison we are also plotting the maximum achievable throughput (i.e. capacity) computed using a theoretical model we developed, and described in an accompanying draft paper. We can see that our XG system is far from achieving the optimal value. In general, there are several factors preventing XG from achieving the maximum. Among them includes fairness (i.e. the optimal solution in the theoretical model gives exclusive access to the shortest links, which may not match the actual traffic demand),

using energy-inefficient modulation, not using long and highly efficient but computationally expensive codes. In this example, the main reason for the difference was our use of energy-efficient modulation.

We assumed bandwidth efficiency of 1 bps/Hz or less, therefore saving energy and extending the network lifetime, an important factor in mobile networks. To corroborate this, we are also plotting the maximum throughput achieved if the waveforms used are restricted to QPSK modulation. As we see, our system is within reason of the maximum achievable by an energy-efficient system for this simple topology. The impact of more complicated topologies on this MAC is the subject of future work.

The above results are representative of the types of studies that can be performed using the XEP. Using the XEP, we have investigated the relative merits of underlay vs. overlay, and the impact of topology control and the interference tolerance of the primaries on the overall network throughput. Preliminary results from these studies are available in the form of a draft report included with the deliverables.

6 Technical Interactions

In support of the DARPA XG program goals of transitioning the technology developed to GOTS and COTS systems, BBN had numerous interactions with the XG working group, industry, academia, and regulatory bodies. We present a brief summary of significant interactions and related activities below.

6.1 *Working group meetings and other briefings to XG performers*

BBN has engaged the XG Working Group participants in the development and refinement of the XG RFCs. We participated in and provided briefings in two working group meetings, one of which was hosted by BBN in our Cambridge, MA facility. The working group process was an integral part of the development and release of the XG RFCs; it provided the primary mechanism through which to capture the concerns of the stakeholders (XG Government and SI team participants).

A notable benefit of the WG process was the generation of policy use cases, which provided concrete contexts for the development of the policy language as well as useful test cases for the policy conformance reasoner software.

In addition we have had several interactions via phone and email with all three SI teams, as well as with Alion Science, in regard to the policy conformance software. In addition, we gave several briefings to a team from Shared Spectrum Corporation that visited BBN for a daylong discussion on implementation and integration of the policy technology.

6.2 *Website and mailing lists*

BBN maintains a website to support the XG working group activities as well as to disseminate to the public, with Government approval, key results of the XMAC and XAP contracts. The policy conformance reasoner software has been made publicly available through this website.

In addition, BBN maintains three mailing lists for XG in order to promote discussions and in order to make announcements related to software and RFC releases:

- **xg:** limited to XG Working Group members
- **xg-panel:** limited to XG Industry Panel members
- **xg-public:** open to the public

6.3 *External briefings and interactions*

FCC: BBN provided a briefing to the FCC in support of DARPA's XG policy technology briefing to the FCC. We received a favorable response, and the audience expressed interest in seeing demonstrations of XG policy technology.

NSF Programmable Wireless Networks: BBN has briefed the academic research community at the NSF Programmable Wireless Networks workshop.

XG Industry Panel: BBN participated in the discussions of the XG Industry Panel in which DARPA briefed the industrial research community on XG policy technology.

Philips Research: We have a technical interchange with a group at Philips Research that is actively doing research in the area of spectrum sharing etiquette, and spectrum measurements in the context of IEEE 802.11 standardization activities.

6.4 Academic collaborations

Northeastern University: We have been working with Prof. Karl Lieberherr and Prof. Ravi Sundaram of Northeastern University in the area of investigating the application of Aspect Oriented Programming techniques in XG systems. Another area of discussion involved the modeling of secondary spectrum markets using truth-telling auction mechanisms. Pengcheng Wu, a student of Prof. Lieberherr, participated in the XAP effort as a summer intern at BBN. Overall, this interaction has positively influenced our development of the XG Abstract Behaviors.

Virginia Tech: We have collaborated with Prof. Amitabh Mishra of Virginia Tech to develop a proposal to the NSF to apply XG policy technology in the context of reuse of 3G cellular spectrum.

Lehigh University: Recently, towards the end of XAP contract, we have had brief email interactions with Prof. Donald Hillman of Lehigh University. The area of discussions included the further development of the XG ontologies (the structure of authority in particular) and the formulation of the dynamic spectrum assignment problem as a deductive database problem. The ideas from these discussions are worthy of further exploration in Phase 3.

7 Summary and Future Work

The current practice of statically allocating spectrum is widely recognized as inefficient. Although virtually the entire spectrum has been allocated to services and much of it is assigned to users, measurements within the XG program and elsewhere have demonstrated that large portions of spectrum are not being used at any given time or location. Regulatory agencies are now considering opening up spectrum for opportunistic access.

Furthermore, while opportunistic access is of tremendous value in the civil sector, it is vital for the military. Since the military is increasingly in need of (and reliant upon) high bandwidth communications, maximizing use of spectrum is an essential military capability.

The primary objective of XG is to increase spectrum utilization by enabling devices to opportunistically utilize spectrum whenever and wherever it is available. A secondary effect of this capability will be to greatly reduce spectrum-planning time prior to deployment. The XG program is developing technologies that fall in two classes:

1. **Spectrum Agility:** Actual spectrum usage varies with location and time. Devices must, therefore, possess *spectrum agility*:
 - a. They must be able to acquire awareness of the spectral environment
 - b. They must be adaptive in the use of spectrum
 - c. They must be able to coordinate the use of spectrum with other devices.

The XG program has developed new technologies and systems that leverage recent advances such as wideband sensing and software radio technology to provide spectrum agility to wireless devices.

2. **Policy Agility:** The use of available spectrum is subject to policy rules that vary with frequency, type of service, type of operator, location, and the regulatory jurisdiction. Furthermore, it is a challenge to manage the accreditation of agile devices for all the uses they are capable of across regulatory regimes. Thus devices that make use of opportunistic spectrum access must possess *policy agility*: They must be able to adapt their operation to opportunities that are authorized by policy, using verifiable approaches. The XG program has, therefore, developed a trusted *policy language framework* to provide policy agility to wireless devices.

7.1 BBN's Accomplishments in the XG Program

BBN Technologies is performing the role of architect in the XG program. BBN has developed a series of XG Requests for Comments (RFCs) that document the vision, architecture, as well as technical specifications for XG systems. Following are a few of our accomplishments during XG Phases 1 & 2:

- Produced four RFCs that define aspects of the XG architecture
- Developed a policy language framework that will eventually enable XG-capable radios to adhere to relevant policies as they dynamically choose where and how to use available spectrum. An additional benefit of this machine-readable policy language is the ability to define “temporary” policies to which radios would conform – thus encouraging experimental use of spectrum.
- Developed reference Policy Conformance Reasoner (PCR) software that demonstrates how a radio system can check whether its intended emissions conform to policy (as specified in the defined policy language). The PCR can also search the policy constraint space for opportunities that are allowed by policy. A version of this software has been delivered to all XG performers

and at least one system integration team has integrated the PCR with their system and demonstrated the combined capabilities in a limited form.

- Produced a simulation environment that allows us to estimate the possible performance gains of various XG spectrum sharing algorithms and protocols.

In addition BBN has supported DARPA's goals of dissemination of key aspects of the XG architecture to the public through briefings at the FCC, NSF Programmable Wireless Networks community, and the XG Industry panel. BBN has also briefed, DARPA personnel from other offices (IPTO), and researchers from the industry (Philips Research), and academia (Northeastern University and Virginia Tech). Furthermore, BBN has worked with the XG system integration teams to support transition of the XG policy technology, by providing access to BBN-developed software as well as through briefings. BBN has maintained a website and three mailing lists for this purpose.

In Phase 2 BBN developed an architectural framework for XG systems, and investigated the core research issues for policy-agile XG operation. We demonstrated proofs-of-concept of the technologies we developed through a prototype policy conformance reasoner software, notional XG policy examples, and basic ontologies populated with concepts that are sufficient for a small range of interesting XG policies. Effectively transitioning the XG policy technology into DoD systems requires further development effort that can directly build upon our results and deliverables from Phase 2.

Milestone	Pre XG	End of Phase 2	Early Phase 3	End of Phase 3	Post XG
Time		Dec 04	Aug 05	Feb 07	Oct 08
Objective Implementation		Demo XG Policy Engine	SI Integration	SI Demo	DoD Transition (JTRS)
Spectrum Policy Representation	English Text	Ontology (XG Notional) 100 concepts 50 policies 200 facts	Ontology (XG Domain) 300 concepts 150 policies 1000 facts	Ontology (XG Domain) 1000 concepts 500 policies 5000 facts	Ontology (DoD Domain) 5000? concepts 2000? policies 20000? Facts
Opportunity Query	Spectrum: None IETF: Point spec, 7 Dimensions	Fully bound spec (a region) 13 Dimensions	Partially bound spec, 1 unbound variable	Partially bound spec, n unbound variables	Utility Spec. 100? Dimensions
Policy Encoding	Text Procedural language Hardware	OWL CLIPS String/socket	OWL CLIPS CORBA	OWL Custom Reasoner CORBA	Web Services Custom Reasoner CORBA Hardware assist
Policy Loading	Built-in	Off-line merge On-line load from local/network store	+ Dynamic Merge	+ Performance enhancements	+ Secure load
Real-time Reasoning	None	Validation (120ms)	Validation (10ms) Search (5s)	Validation (10ms) Search (1s)	Optimization (10s) Search (100ms) Validation (1ms)
Accreditation	Case by Case Policy sets X platforms	Architecture reduces the number of accreditations		Accreditation roadmap	Policy sets + Platforms + Reasoner

Figure 21: XG Policy Technology Roadmap

As the XG program enters system integration, test, and demonstration in Phase 3, we have developed a roadmap for XG policy technology, in consultation with the DARPA PM for XG. The roadmap, illustrated in Figure 21, summarizes the current status of the technology, and traces the development needed in Phase 3 and beyond. The key milestones on the roadmap for this technology in Phase 3 are integration and demo with XG systems. The integration and eventual transition to DoD systems must be accompanied by further development of the language framework, in particular, through population of

the policy ontologies with a larger repertoire of relevant concepts, and the creation of tools suitable for policy administrators and defense spectrum planners.

We have also converged on a proposed architecture for XG system integration during Phase 3 illustrated in Figure 22. This architecture clearly identifies the relationship of the policy engine (shown in green) within the XG system – we believe that BBN’s XG policy conformance reasoner technology fits cleanly into this architecture. Also note that the clean architectural separation of policy from the system enables changes and innovations in either side to proceed independently.

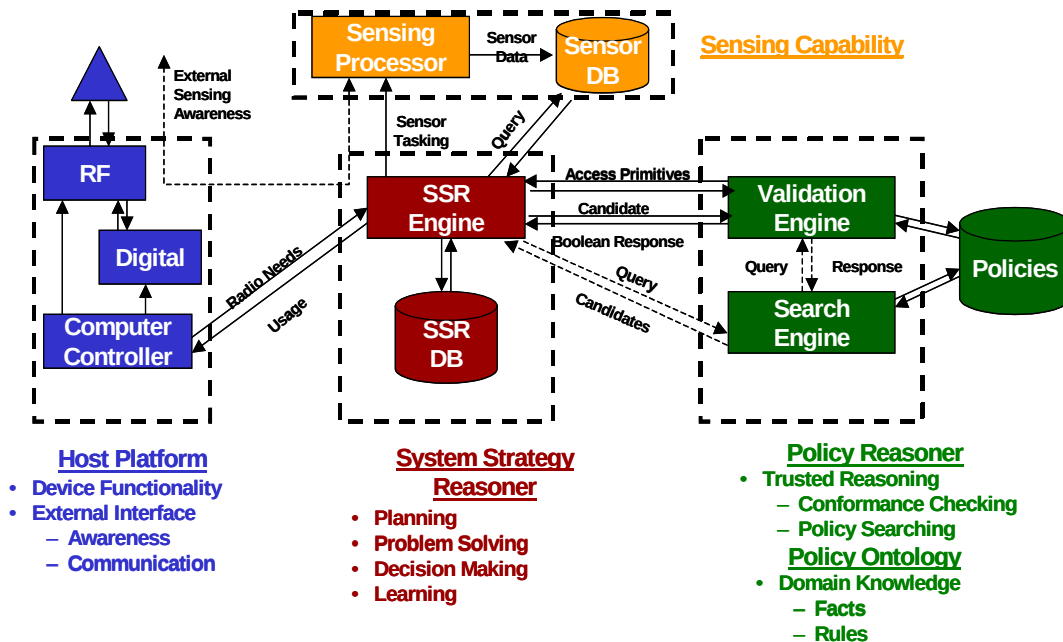


Figure 22: XG System Integration Architecture

This architecture suggests that it would be desirable to develop additional primitives to access the sensor database and integrate them with the policy conformance reasoner.

7.2 Future Work

Follow-on work that builds on our existing work will be required in order to support the XG program in Phase 3 and beyond. We believe the following tasks are critical during Phase 3 of the XG program:

- **Policy Conformance Reasoner (PCR) Software Enhancements:** The current policy conformance reasoner (PCR) software was developed as a set of functional prototypes. It includes an initial release of the Policy Validation Engine (PVE) software and an early proof of concept for the Policy Search Engine (PSE). Both of these systems require further development to add robustness, provide programmatic interfaces, and demonstrate substantial performance enhancements as we make them ready for integration into a target XG system. Furthermore, they must be tailored to the requirements of the particular XG system platform of the down-selected Phase 3 system integration team.
- **Policy Language and Content Representation Framework Enhancements and Authoring Support Tools:** As the XG policy technology leaves the laboratories of the XG performers and is transitioned into DoD systems, it is critical that basic tools exist to enable domain users such as system administrators and spectrum managers to create, manage, check, and provision policies for XG systems. Templates for common policy

idioms can make it easier for policy creation and management. Furthermore, the ontologies must be populated with enough content, and reorganized for more processing efficiency if needed, so that a wide range of useful policies for XG systems can be readily expressed.

In addition, the following important tasks must be considered before XG technology would be ready for transition and deployment into DoD networks.

- **Certification Roadmap:** Before XG systems can be deployed in the field - both within the US and in other countries - they must be certified for use by the relevant authorities. A study, possibly modeled on prior work on the certification of software defined radios, is required to identify the issues involved in certifying XG systems.
- **Embedded PCR Study:** A study is needed to analyze the feasibility and benefits of building a reasoning and inference engine in low-cost embeddable hardware instead of software. An embedded implementation may potentially offer performance, cost, or ubiquity benefits.
- **Security and Trust Model Evaluation:** A study is required to investigate security and trust threat models for XG policy. While XG radios are inherently as secure as software radios and their host OS and networking stacks, we need authentication to ensure that the rules for how the radio may operate in different conditions are indeed the policies issued by the responsible authorities.
- **Network Strategy Reasoner:** As policy-agile XG systems are deployed and integrated into the network-centric force, a study on how to extend XG methodologies to the realm of a dynamic network (beyond just spectrum) management will be beneficial.

References

1. [XGV] The XG Vision RFC
 2. [XGAF] The XG Architectural Framework RFC
 3. [XGAB] The XG Abstract Behaviors RFC
 4. [XGPL] The XG Policy Language Framework RFC
 5. [XEP] The XG Evaluation Platform software documentation
 6. [PCR] The XG Policy Conformance Reasoner software documentation
 7. [SPTF] Spectrum Policy Task Force Report, FCC ET Docket no. 02-135, November 2002,
http://hraunfoss.fcc.gov/edocs_public/attachmatch/DOC-228542A1.pdf
- [DP] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, Design Patterns: Elements of Reusable Object Oriented Software, ISBN: 0201633612, Addison-Wesley, 1995.