Cognitive Radio

An Integrated Agent Architecture for Software Defined Radio

Dissertation

Doctor of Technology

Joseph Mitola III

Royal Institute of Technology (KTH)

Teleinformatics

Electrum 204

SE-164 40 Kista

Sweden

TRITA-IT AVH 00:01

ISSN 1403-5286

ISRN KTH/IT/AVH—00/01--SE

This dissertation is submitted in partial fulfillment of the degree of Doctor of Technology.

8 May, 2000

Abstract

This dissertation addresses the intersection of personal wireless technology and computational intelligence. The primary research issue addressed is the organization of radio domain knowledge into data structures processable in real-time that integrate machine learning and natural language processing technology into software radio. The thesis defines and develops the cognitive radio architecture. The features needed in the architecture are derived from cognitive radio use cases. These include inferring user communications context, shaping access-network demand, and realizing a protocol for real-time radio spectrum rental. Mathematical foundations for the knowledge-representation architecture are derived by applying point-set topology to the requirements of the use cases. This results in the set-theoretic ontology of radio knowledge defined in the Radio Knowledge Representation Language (RKRL). The mathematical analysis also demonstrates that isochronous radio software is not Turing-computable. Instead, it is constrained to a bounded-recursive subset of the total functions. A rapid-prototype cognitive radio, CR1, was developed to apply these mathematical foundations in a simulated environment. CR1 demonstrated the principles of cognitive radio and focused the research issues. This led to an important contribution of this dissertation, the cognitive radio architecture. This is an open architecture framework for integrating agent-based control, natural language processing, and machine learning technology into software-defined radio platforms.

Dedication

For Lynne'

Acknowledgement

First, I would like to acknowledge the support and advice offered by Professor G. Q. ("Chip") Maguire of KTH without whose initial enthusiasm the research would not have been undertaken. Without his tireless and patient mentoring, it might not have been completed. Dr. Erland Wikoburg of my advisory committee kept me thinking about the physics and real-world implications of my research. Professor Jens Zander threw down the gauntlett "its too long" a couple of times. I still take too many words to say things, but now I'm more aware of the value of brevity. Dr. Svante Signall, my Licentiate Opponent, wondered if the doctoral research could be completed "in such a brief amount of time" – seems like forever and only yesterday. Theo Kanter tackled the equally important network side of things with his work in active memory. Working in the same laboratory as Theo has been stimulating. In addition, special thanks are due to Professor Dr. re. nat. Friedrich Jondral for graciously agreeing to serve as opponent for the dissertation defense at the time that he was organizing and conducting his own software radio workshop at Karlsruhe. It has been a distinct pleasure and enlightening experience to interact with Chip, Theo, and other professors, graduate students and researchers at KTH who are blazing trails for the future of wireless technology.

The US Navy sponsored the modeling and simulation aspects of this research under MITRE contract number DAAB07-99-C-C-201. Special thanks to Rosemary, Allan, and Bob. In addition, I would like to acknowledge the climate of interest and support at MITRE created by Dr. Bob Mikelskas. Bob supported my participation in MITRE's accelerated graduate degree program, which also supported part of this research.

Finally, this work is dedicated to my wife, Lynne' in recognition of her initial enthusiasm for the undertaking and of her unending love, patience, understanding and support during the two a half years this work was in the making.

Brief Glossary

Antecedent	The pre-condition of an If-Then logic structure
3G	Third Generation mobile cellular systems, e.g. based on CDMA
CIR	Carrier to Interference Ratio, a measure of radio signal strength
CORBA	Common Object Request Broker
ETSI	European Telecommunications Standards Institute
HTML	Hypertext Markup Language
IDL	CORBA Interface Definition Language
ITU	International Telecommunications Union
Method	A procedure attached to a software object
Ontology	The branch of metaphysics that studies the nature of existence or being
OOT	Object-Oriented (OO) Technology
ORB	Object Request Broker, software that dispatches remote procedure calls, etc.
PDA	Personal Digital Assistant
PDANode	An object that contains an srModel and related slots and methods.
Reinforcement	Assigning an increased degree of belief or relevance to a stimulus, plan, etc.
RKRL	Radio Knowledge Representation Language
SDL	The Specification and Description Language (ITU Recommendation Z.100)
Slot	A data structure that is part of a software object
srModel	Stimulus-Response Model
Taxonomy	Science dealing with description, identification, naming, and classification
UML	Unified Modeling Language, an object-oriented design language
W-CDMA	Wideband CDMA, the emerging international 3G standard
XML	The eXtensible Markup Language

Table of Contents

1 Introduction

This dissertation defines and develops cognitive radio, the integration of model-based reasoning with software radio [1] technologies. It analyzes the architecture and performance of a rapid-prototype cognitive radio, CR1 in a simulated environment. This architecture is based on the set-theoretic ontology of radio knowledge defined in the Radio Knowledge Representation Language (RKRL) [2]. CR1 incorporates machine-learning techniques to embrace the open-domain framework of RKRL. These machine learning techniques make the software-radio *trainable* in a broad sense, instead of just programmable. Although somewhat primitive, CR1's level of computational intelligence provides useful insights into the research issues surrounding cognitive radio. CR1 integrates aspects of digital signal processing, speech processing, theory of computing, rule-based expert systems, natural language processing, and machine learning into the software radio domain. The inter-disciplinary nature of cognitive radio raises interesting questions for future research and development.

1.1 Overview

This dissertation addresses the intersection of personal wireless technology and computational intelligence. The term cognitive radio identifies the point at which wireless personal digital assistants (PDAs) and the related networks are sufficiently computationally intelligent about radio resources and related computer-to-computer communications to:

(a) detect user communications needs as a function of use context, and

(b) to provide radio resources and wireless services most appropriate to those needs.

The results are derived in part from the development of a rapid-prototype cognitive wireless PDA, CR1, that uses structured computational models of services and radio etiquettes to control the delivery of next-generation wireless services in a simulated environment. The computational models underlying cognitive radio include RKRL, reinforced hierarchical sequences, and the cognition cycle [3]. RKRL 0.3 consists of forty micro-worlds described in the Extensible Markup Language (XML) and summarized in this thesis. Reinforced hierarchical sequences organize CR1's internal representation of itself, its user, and its environment, including the radio networks accessed through its software-radio host platform. The architecture

is somewhat biologically inspired, based on neural-network-like nodes that respond to external stimuli and that process the resulting internal data structures. These structures support a cognition cycle consisting of Observe, Orient, Plan, Decide, and Act phases. Machine learning is integrated throughout the architecture. Its waking behavior includes incremental learning. Its sleeping behavior provides for batch learning, and its prayer behavior supports supervised learning. The CR1 Java environment includes a design language by which the CR1 prototype was constructed from a palette of node classes. CR1's wake cycle is optimized for the rapid delivery of wireless services, but it supports the observation of stimuli needed for subsequent machine learning. In the sleep cycle, stimuli that were previously not experienced are integrated into the internal node network. In the prayer cycle, unresolved conflicts may be presented to the designer for resolution. After initial training, CR1 has a capability to be further trained either by the user or by the network. In addition, it may be initialized to a previously learned starting point, a personality. The personality consists of sets of internal models structured according to RKRL. The same machine learning algorithms that enable learning from text and speech streams allow the system to recognize external context and to process radio protocols. The resulting equivalence of human-human, human-machine, and machine-machine protocols evident in CR1's machine learning over reinforced hierarchical sequences seems under-explored in data communications science and engineering.

RKRL was initially defined via use-case scenarios [2]. The use cases were analyzed via simulations. One use-case, described in detail in Appendix D [4], shows how the properties of RKRL support a new etiquette for the real-time pooling of radio spectrum. Over-the-air downloading of such radio etiquettes raises questions about the stability and other computational properties of agents like CR1 and languages like RKRL.

The computational properties are analyzed using the recursive function theory of computing. The theorems provide the foundation for cognitive radio to reason about its own computational capabilities. This includes reasoning about processing capacity and memory required for the execution of new functions downloadable from a network. RKRL defines a formal meso-world that includes the micro-worlds of time, space, location, radio resources, protocols, communications context, and services to users. Its meta-world defines the core concepts in terms of which the other micro-worlds described. For example, its meta-world

defines set-theoretic concepts like "contains," the set-membership ontology. The other micro-worlds grow to encompass additional knowledge (e.g. of an new protocol) using the formal "contains" model. Micro-worlds embedded in a PDA may also atrophy if knowledge is not used. For example, detailed knowledge of a GSM-derived PCS mode may be forgotten by a cognitive PDA that is using IS-95 instead. Such a PDA can always access that knowledge from a cognitive host network, however, because the core ontology retains the fact that the GSM-based PCS mode is known by the network. RKRL is fairly comprehensive, consisting of about 4000 model-based knowledge-representation frames.

CR1, on the other hand, is highly focused, employing just that subset of RKRL necessary to establish the findings of this dissertation. The purpose of creating the CR1 rapid prototype was to test RKRL and the concepts of cognitive radio that were initially developed in my Licentiate thesis. This required integrating elements of machine learning, natural language processing, and software radio into a mutually supportive computational framework. The development and related analysis led to the formulation of the cognitive radio architecture.

Software radio provides an ideal platform for cognitive radio. Although cognitive radio algorithms can control a programmable digital radio, software radio and the emerging complexity of third-generation wireless (3G) systems motivate the development of cognitive radio. This includes assisting users in dealing with the exponentially increasing array of wireless access methods.

1.2 Organization of this Thesis

The next chapter reviews background necessary for cognitive radio and defines prerequisite technology, particularly the software radio. A more complete discussion of the architecture of software radio may be found in the author's text *Software Radio: Wireless Architectures for the 21st Century* (alpha edition, to be published in an expanded form in September, 2000 by Wiley Interscience [1]). Chapter 3 summarizes relevant prior research. Chapter 4 defines cognitive radio and introduces four key areas in wireless communications on which cognitive radio will have an impact. These are radio resource management, network management protocols, services delivery, and type-certifiable downloads. The use cases of Chapter 5 statistically characterize performance aspects of cognitive radio. RKRL 0.3 as used in

the creation of CR1 is then defined in Chapter 6. Chapter 7 provides an overview of the rapid prototype, CR1. Chapter 8 consolidates the conclusions of this work in an initial cognitive radio architecture. Implications of the research are discussed in Chapter 9.

The thesis concludes with relevant appendices. Appendix A: "Software Radio Architecture Evolution: Foundations, Technology Tradeoffs, and Architecture Implications" (Invited Paper) describes the technology evolution of software radio from the author's contribution to the IEICE, Japan, Transactions on Communications, Special Issue on Software Radio Technology. Appendix B: Software Radio Architecture: A Mathematical Perspective provides the mathematical foundation for cognitive radio's ability to reason about its internal computational resources. This core contribution shows that isochronous systems like software radios cannot benefit from Turing-computability, but instead must operate within a subset of the Turing functions, the bounded recursive functions. The guarantees of stability of this subset of Turing-computability make it possible for cognitive radio to reason effectively about its own internal structure without violating Goedel's incompleteness theorem. This appendix is a reprint of the author's paper from the IEEE Journal on Selected Areas in Communications. Appendix C is a reprint of: "Cognitive Radio: Making Software Radios More Personal." This version of the author's paper for the IEEE PCS Magazine (August 99) provides a stand-alone overview of cognitive radio and RKRL. Appendix D: Cognitive Radio for Flexible Mobile Multimedia Communications, defines the pooled spectrum air interface enabled by cognitive radio. This paper won the "best student paper" at the IEEE 1999 Mobile Multimedia Conference (MoMuC, November, 1999). Appendix D is the paper "Cognitive Radio: Agent-based Control of Software Radios." This paper was presented at the 1st Karlsruhe workshop on software radio technology, Karlsruhe, Germany on 29 March, 2000. Finally, the softcopy version of RKRL 0.3, written in XML is the on-line companion to this thesis.

2 Background

This chapter provides technical background that is essential to the thesis. It includes a review of the technology for defining air interfaces and protocols. This review characterizes the use of formal methods in the standards development process. It yields both a list of computer languages for representing radio knowledge and some motivation for RKRL. This chapter also includes a definition of software radio, including the modeling of the software radio architecture. It is shown that a language that consistently keeps track of computational resources enables the autonomous creation of software that is provably stable. Such software will not use computing or communications resources that exceed tight upper bounds, except in the case of a hardware failure. This chapter summarizes the state of the art in location- and environment-aware computing, which cognitive radio seeks to extend. Since machine learning and natural language processing are essential to cognitive radio, this chapter also provides background on these technologies with examples that provide a glimpse ahead into the core technical contributions of this thesis. The companion chapter on prior work addresses related research that bears on cognitive radio, while this chapter defines its foundations.

2.1 Air Interface- and Protocol-Definition Processes

The process of extending mobile units and radio infrastructure to new physical layer capabilities and protocols is labor-intensive and prone to error. Natural language specifications are sometimes easy to misinterpret. Latent ambiguities may be identified during the product development process, and sometimes not until after systems have been deployed. This section provides an overview of the migration towards the use of formal languages and modeling methods to improve this specification process. This topic is essential background for cognitive radio and RKRL since many of the approaches now in use are relevant, but fall short of either the expressive power or mathematical structure necessary for cognitive radio.

An early thrust to reduce the ambiguity of specifications of communications systems resulted in the Specification and Description Language, SDL [5]. This language clearly represents state machines and message sequences that are the cornerstone of radio protocols

(Figure 2-1). The International Telecommunication Union (ITU) adopted SDL in its Z.100 recommendations. In addition, the European Telecommunications Standards Institute (ETSI) recently adopted SDL as the normative expression of radio protocols.

> **"SDL [shall be] the normative specification of behaviour within a standard with text supporting and clarifying the requirements expressed in the SDL." [6].**

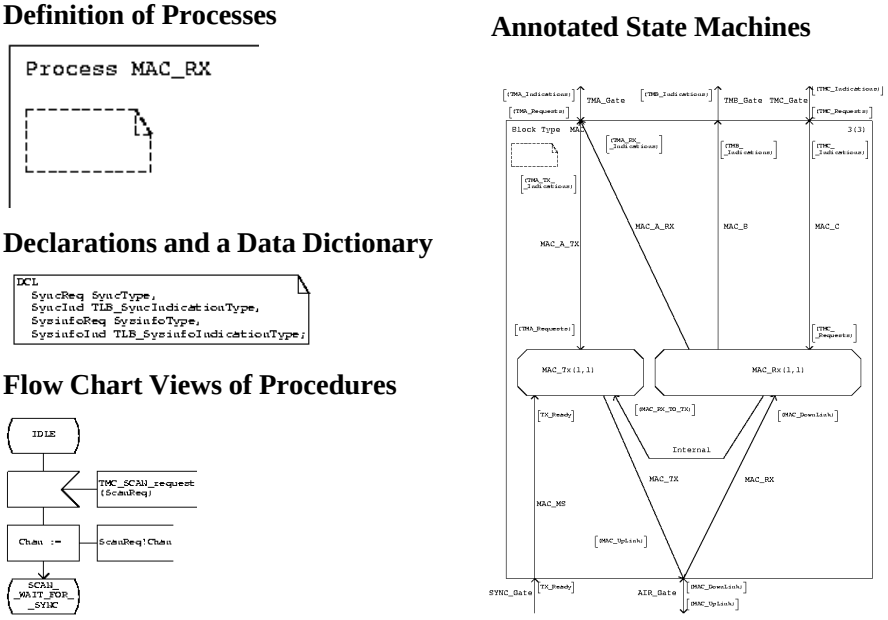


Figure 2-1 Highlights of SDL

This suggests continued movement towards computer-based formal models in the specification processes. One thus expects SDL modeling of radio to continue to expand. SDL, however, lacks tools that support general-purpose object-oriented modeling. Such techniques have shown significant benefit in the creation of real-time telecommunications systems [7, 8, 9].

There has been a proliferation of object-oriented modeling techniques [10]. Recently, however, the Unified Modeling Language (UML [11]) resulted from the unification of diverse object-oriented analysis, modeling, design, and delivery methods. UML readily expresses software objects, including attached procedures ("methods"), use cases, and the packaging of software for delivery. In principal, UML can model anything. In practice, it is useful in software design and development, but it is weak in the modeling of hardware devices and

databases. UML extends the reach of formal modeling pioneered by SDL into areas such as user interfaces. UML also facilitates the formal modeling of abstract radio concepts such as a GSM (Global System for Mobile communications) Slow Access Control Channel (SACCH). A SACCH is a virtual channel that is a subset of a low rate time-division multiplexed channel which itself is a time-division multiple access burst channel within a 200 kHz RF channel [12]. UML readily expresses the data structures associated with an SACCH. The Software-Defined Radio (SDR) Forum, furthermore, has adopted UML as its method for representing open-architecture radio [13].

On the other hand, although UML can provide a framework for radio system design, specialized radio domains such as propagation modeling need application-specific propagation modeling tools [14]. These are historically written in C or FORTRAN. The need to integrate such diverse tools into a seamless user environment led to the development of the Common Object Request Broker Architecture (CORBA). CORBA defines a framework for integrating software packages using an Object Request Broker (ORB). The ORB "middleware" transforms what amount to remote procedure calls from a client application into a format acceptable to the server application. The transformations of the procedure calls are defined using the CORBA Interface Definition Language IDL [15]. Using CORBA, one can readily combine an air interface written in C on a Digital Signal Processor (DSP) with a propagation-modeling tool written in FORTRAN. Before the development of multiband multimode mobile radios, there was little need for the integration of such diverse software packages. The use cases of this thesis show how that situation is changing.

Although SDL, UML, and CORBA/IDL provide important new capabilities, all of these languages are lacking in important ways. None of these languages offers a standard reference-ontology. That is, if one is designing a new third generation (3G) air interface using SDL, one must rely on the informally defined notions of "channel", "frequency", "multiple access," etc. The more generic the term (e.g. channel), the more likely that its use in one context (e.g. Digital European Cordless Telephone - DECT) is different from its use in another context (e.g. 3G). Yet, these concepts are central to defining a commercial-grade air interface. SDL and UML allow one to register these concepts in a data dictionary, but the semantics are left up to the user. One could argue that this is as it should be. Given a general purpose tool like UML, one can

informally coordinate the ontological perspectives using the coordination processes of the ITU, ETSI, etc.

On the other hand, other industries are beginning to employ reference ontologies. Chemical engineers, for example, are using the Chemical Markup Language, CML, to exchange chemical formulas using an application-specific reference ontology [16]. CML is based on XML, which can be thought of as Hyper-Text Markup Language (HTML) with user-defined tags. In addition, HTML tags define display format, while XML tags define data exchange format. The potential benefits of a standard reference ontology include a more efficient standards-setting process, more structure in the exchange of radio-science data among researchers, and more flexibility in the structured interchange of radio knowledge among mobiles and networks. The benefits of a standard reference ontology for radio have been argued [2], and the SDR Forum began work in February, 2000, on the use of XML for radio domain management [17].

In addition, none of these languages support independent axiomatic treatment. Expressions must be mutually consistent, but they need not have formal logical structure. In this case, there are no pre-defined logical sorts. There are also no axioms to which expressions of the language conform. In addition, they need not be consistent with prior knowledge like a reference ontology. One can attempt to make them consistent with prior knowledge by incorporating the prior UML models into the current model, but this generally leads to context-dependent differences that require substantial rework. RKRL, on the other hand, defines an axiomatic framework. It provides the cognitive radio reference ontology of world knowledge, radio protocols, air interfaces, networks, and user communications states. It also provided the knowledge-organization framework for the rapid prototype, CR1. Since it is written in XML, it could also help in communicating a wide range of radio-technical knowledge among people and among future RKRL-capable software agents.

2.2 The Software Radio, PDR and SDR

A software-defined radio (SDR) is a radio that can accommodate a significant range of RF bands and air interface modes through software [18]. For the ideal software radio, that range includes of all the bands and modes required by the user/ host platform. Appendix B precisely

defines the software radio, differentiating SDR from software radio. This section provides a synopsis focused on cognitive radio. The first sub-section defines the software radio and its variants, while the second sub-section summarizes the architecture-level model of the software radio for cognitive radio. Finally, computability issues are addressed.

2.2.1 Defining the PDR, SDR, and Software Radio

Military radios for mobile ground, air and naval applications employ the bands HF through UHF. Thus, 2 MHz to 2 GHz defines the nominal RF coverage of the tactical military radio. Commercial mobile radios typically do not require the HF band, but emphasize the mobile satellite, cellular and PCS bands between 400 and 2400 MHz instead. In the limit, an ideal mobile military software radio digitizes 2.5 GHz of RF, sampling at 6 giga-samples per second (gsps) with at least 60 dB of dynamic range, including the contributions of automatic gain control (AGC). The state of the art is 6 gsps and 30 dB [19]. In addition, the 6 gsps DACs and filters cannot yet reach the spectral purity needed for the ideal software radio. With today's technology, then, one must compromise the ideal, implementing programmable digital radios (PDRs), also called software-defined radios (SDR). An SDR, for example, might use a fast-tuning synthesizer to hop a 3 MHz instantaneous baseband bandwidth over a 200 MHz agility bandwidth. Although the pattern of hops generated by the synthesizer could be programmable, the instantaneous waveform bandwidth is limited to 3 MHz. A software radio, on the other hand, would digitize the 200 MHz bandwidth at 400 msps or more. The instantaneous bandwidth of the waveform, then, could be programmed to any bandwidth up to 200 MHz, not limited by the 3 MHz baseband. The PDR would have a programmable choice of hopping frequencies across 200 MHz, while the software radio would have an *arbitrarily* programmable waveform across the 200 MHz[1].

The degree of flexibility of a radio platform is of considerable relevance to cognitive radio. A commercially viable SDR may be implemented as a mix of Application-Specific Integrated Circuits (ASICs), Field Programmable Gate Arrays (FPGAs), Digital Signal Processors (DSPs),

1 Of course, the rest of the architecture such as antenna, RF conversion, DACs, intermediate buses, signal generation speed, and the processing capacity of the receiver architecture also have to support the bandwidth of the ADC in a viable implementation. That is, the rest of the system should be compatible with the bandwidth of the ADC.

and general-purpose microprocessors. ASIC parameters also may be defined in software. But, an ideal software radio has no ASICs. Instead, it implements all the radio's RF conversion, filtering, modem and related functions in software. The range of SDR implementations as a function of digital access bandwidth and processor programmability is shown in Figure 2-2.

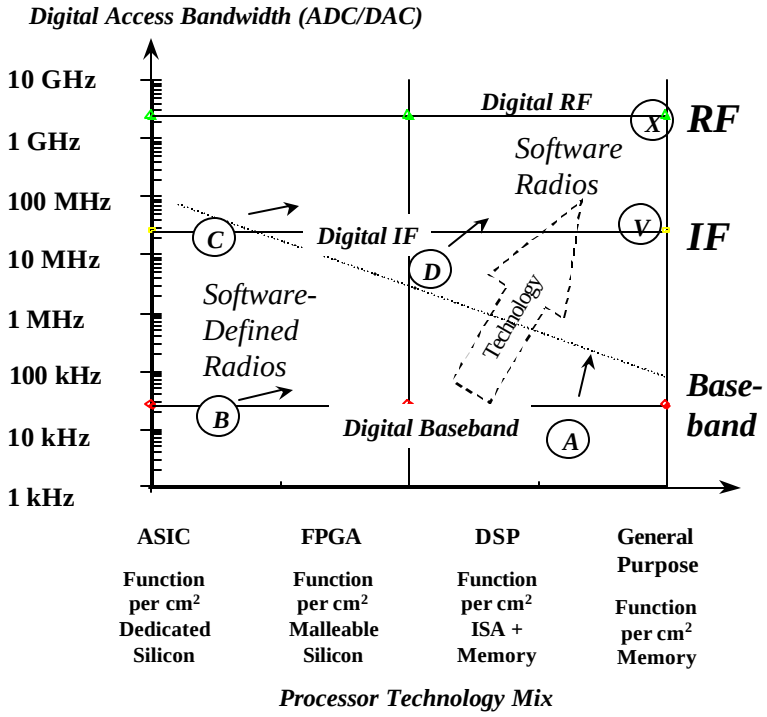


Figure 2-2 Dimensions of Software Radio Implementations

Implementation points A through D in the figure are contemporary SDRs (see Appendix A). The Virtual Radio [20], however, is an ideal single-channel narrowband software radio based on a general-purpose processor, specifically a DEC Alpha, running UNIX. Point X is the ideal software radio with digital access at RF and all functions programmed in general purpose processors. Although providing maximum flexibility and thus of research interest, such designs are economically impractical in the short term. Cognitive radio's focus, however, is on the fourth generation of wireless in which commercial radios will begin to approximate the software radio.

In the commercial sector, for example, a dual antenna-RF stage could cover the radio spectrum from 350 MHz to 2400 MHz. With a 100 MHz x 16 bit ADC and a few GFLOPS of

processing capacity, this would be a powerful software radio precursor.

2.2.2 Modeling the Software Radio

The architecture-level model of Figure 2-3 identifies the functional components and interfaces of the software radio (see Appendix B). In order to reason about its own internal structure, a radio requires some such model. Cognitive radio's internal model of itself is based on this specific model.

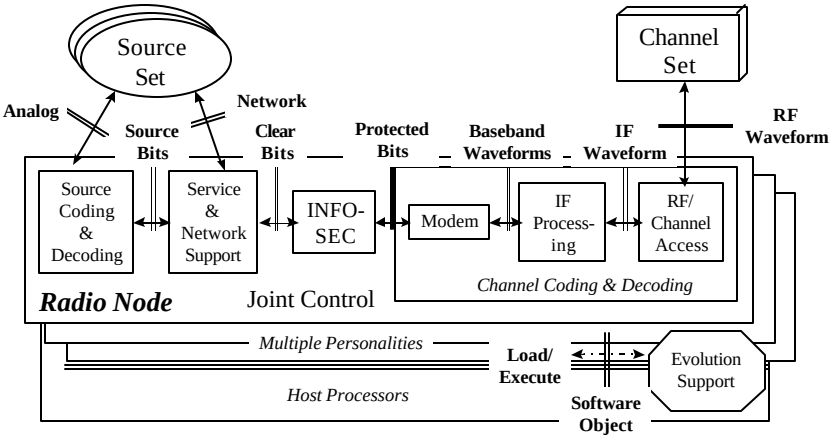


Figure 2-3 Architecture Model of the Software Radio [139]

There are, however, many other comprehensive, precise models of the internal architecture of a radio. The architecture model of the SDR Forum, for example, augments the modem function with a data processing function on the channel coding and decoding side of the INFOSEC module. In addition, their model has no IF Processing module. One can create a mapping among the two models that establishes their equivalence. The appendix develops a topological model within which such mappings may be analyzed. That analysis includes the question of constraints on computational resources in software radio.

2.2.3 Computational Resources

Software radio includes the downloading of software objects that modify or extend the host radio's capabilities. The SDR Forum, for example, has defined a secure download protocol that assures the object is suited to the host radio, appropriately authorized, and free of error [21]. The thinking is that a new well-behaved object that interacts with existing well-behaved objects in the radio will yield a well-behaved system. Appendix B addresses the question "under what

conditions can well-behaved radio software be composed with other well-behaved radio software to yield a well-behaved system?" This general question of software composition is known to be undecideable [22]. That is, there is no algorithm that can examine two arbitrary pieces of software in finite time and yield "Yes" if they will yield an answer and "No" if they will consume infinite resources (and thus yield no answer). Fortunately, software radios are engineering systems with timing constraints that allow one to prescribe constraints on the topological structure of the software that establishes conditions under which composition of software modules is well behaved [139]. Since modems, equalizers, vocoders, and other radio functions are part of an isochronous stream, they must run-to-complete in a short time window that a radio engineer can bound tightly in advance. There are therefore a-priori bounds on time and space (i.e. memory) for SDR modules that each new module must meet as well. Theoretically, these bounds comprise a step-counting function, a function that counts resources used by another program.

There are programming constructs that will meet these bounds sometimes, but not for all possible circumstances. In particular, any construct that is equivalent to a while or until loop (including recursions) can cause two well-behaved software modules (or objects) to consume infinite resources in unpredictable ways. Appendix B on mathematical perspectives shows that one may constrain while and until loops into bounded-resource equivalents (bounded-while and bounded-until). These functions will consume up to a specified amount of memory and time independently and up to another specified amount in concert. The theorems of the appendix show that the bounded recursive functions are the largest set of functions for which predictably finite resource consumption may be guaranteed. In addition, the composition of bounded recursive functions is also bounded recursive, and this sets measurable conditions under which plug-and-play modules will not use excessive resources. Furthermore, there is no practical limit on the useful radio functions that can be computed with the bounded recursive functions[2].

2 One might initially think that this violates Goedel's incompleteness theorem, which says that any self-referential system cannot be defined everywhere and thus can consume infinite resources on some combinations of inputs. "Step-counting functions" (e.g. clock to measure time) can be defined everywhere, so if an application consumes unacceptably large resources, it can be stopped by the violation of a bound on the step-count, restoring Goedel's theorem. This construct converts a potential crash into a mere error condition for reliable counting step clocks (which they are, alone and backed up).

2.3 Environment-Aware Computing

Cognitive radio increases the awareness that computational entities in radios have of their locations, users, networks, and the larger environment. Recent progress in location- and environment-aware computing is therefore relevant to cognitive radio. The European Community sponsored the Advanced Communications Technology and Services (ACTS) program during 1996-99, which included a thrust in location-aware computing. The OnTheMove thrust resulted in mobile middleware ("MASE") for small multimedia terminals [23]. The middleware running on the network down-scales the information content to the format acceptable to the handheld terminal. Subsequently the Wireless Applications Protocol (WAP) commercialized the approach taken by MASE. A City Guide was implemented in MASE [24]. The researchers reported high consumer interest in mobile email, a personal news service, personal stock portfolio, and the city guide map application. Other OnTheMove researchers reported interest in mobile streams (e.g. video), clips on demand (video or audio), and fast file downloads [25].

Beadle, Maguire, and Smith [26] take these notions a step further. They have investigated the use of highly mobile systems to create environment-aware computing and communication systems. Their systems include badges that use wireless communication and sensor technology to become aware of the immediate environment including doors, heating, ventilation, air conditioning, lighting, appliances, telephones, and pagers. Thermal, audio, and video sensors integrated with a PDA yield a new array of opportunities for human-centric information services. This high degree of environment sensing takes mobile computing beyond location-awareness to environment-awareness.

Cognitive radio is based on the premise that OnTheMove, environment-aware computing, and related research will create innovative applications that can use location and other aspects of the environment to tailor services to users. These applications will need a standard world ontology if diverse applications are to smoothly and efficiently share knowledge about the environment. In addition, applications developers will not be expert in each of the myriad of bands and modes available to emerging software radios. Thus, there will be a need for agent-based applications that delegate the control of the software radio to an agent that has been specifically designed for that purpose. The cognitive radio is that agent. The use of knowledge-

based expert systems technology to create such agents leads to knowledge-engineering bottlenecks and software of limited flexibility. Cognitive radio therefore embeds machine learning techniques that enhance robustness to rapidly-changing external environments.

2.4 Machine Learning

In general, machine learning includes supervised and unsupervised approaches to extracting knowledge from raw data [27, 28]. Supervised techniques entail the a-priori association of a class, category, or concept with exemplar data sets. These may be presented as positive or negative examples. Unsupervised techniques, on the other hand, require the algorithm to extract knowledge using general principles such as distance among points in a metric space [29]. Document clustering, for example, maps word vectors into $\mathbf{R}^{\mathbf{N}}$, N-dimensional real space, using information-theoretic metrics [30]. Unsupervised learning is also feasible in symbolic spaces including databases and predicates, e.g. using conceptual clustering [31].

Machine learning is an extensive technology area embracing automatic pattern recognition [32], abductive inference [33, 34] automatic rule acquisition [35], neural networks [36], reinforcement learning [37, 38], and genetic algorithms [39] as highlighted in Figure 2-4. The early pattern recognition literature was concerned with representing features in metric spaces to facilitate clustering using statistical methods. Non-parametric and non-linear metrics emerged throughout the 1970's and 80's. In order to cope with domains in which entities exhibit non-stationary statistics (e.g. speech), sequential analysis methods were developed. The most widespread of these is probably the Hidden Markov Model (HMM). An HMM induces a new state when the current sequence of data is better explained by a change of state than by a change of parameters of a known state. HMM's thus partition multi-modal distributions into their sequential modes instead of forcing the random process into a unimodal distribution that fits the data.

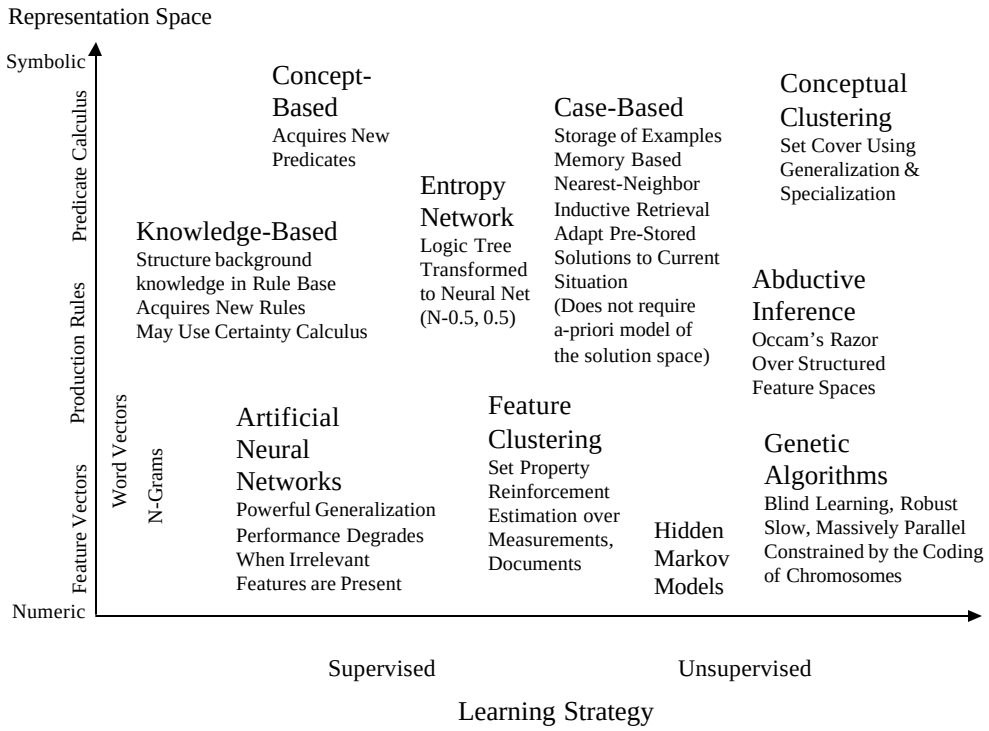


Figure 2-4 Overview of Relevant Machine Learning Techniques

Machine learning has been used in telecommunications, e.g. in call-admissions control algorithms. For example, Q-learning is a form of reinforcement learning. Q(s, a) is the value of action a in state s. A Q-learning iteration has the form:

$$\mathbf{Q_{k+1} = (1-g(s,a))Q_k(s,a) + g(s,a)f(s,a,(\max|_b(Q_k(s,b))}$$ [40].

The weight $g(s,a)$ determines the degree of exploitation of current knowledge, while f shapes the search for new knowledge. As the number of iterations approaches infinity, Q approaches the optimal dynamic-programming policy with probability 1 [41]. This learning technique has been applied to call admissions control in simulated networks [42] and robot control [43]. The literature has many similar algorithms from state-space based automatic control, fixed-point algorithms, Kuhn-Tucker optimality, and genetic algorithms. These address network applications like call admissions control, and do not address user-specific symbolic knowledge like the user's itinerary or the change of a daily commuting pattern. The latter is the focus of cognitive radio.

This section introduces those aspects of machine learning used in this thesis. Since the focus of this thesis concerns radio engineering, the treatment of machine learning is necessarily

high level.

2.4.1 Learning via Extensible Concept Models

Cognitive radio offers many opportunities for machine learning. A user's interaction with a PDA, for example, includes opportunities for both unsupervised and supervised learning. The PDA may develop a model of its time-space pattern of use of wireless resources autonomously, e.g. to reduce the uncertainty of demand offered to the network. This could then be used pro-actively to determine the availability of services historically needed in specific use-contexts. The PDA can also passively observe user actions, e.g. composing and sending wireless email while moving from home to work via a train. Subsequently it might observe the user sending email on the company's RF LAN. It could then ask the user if it should expect to send wireless email via the company's RF LAN when available rather than GPRS or 3G even though both are accessible. Such knowledge exchanges could occur in a mixed-initiative dialog [44]. Acquisition of knowledge from a network, on the other hand, could be rote learning supervised by the network. This thesis explores these possibilities.

Core machine learning technology entails the extraction of a description of a concept from examples and background knowledge [45]. The machine learning process may also produce an algorithm that can recognize additional instances of the learned concept. The notion of a concept is central to machine learning. Concepts exist in a knowledge representation space such as a generalization hierarchy (e.g. eagles are birds which are vertebrate animals [45]). The knowledge representation space for cognitive radio defined in RKRL includes the relevant components of radio hardware (e.g. antennas, ASICs, processing, memory), the user, the network, and the subset of (space * time) in which the radio is used. As illustrated in Figure 2-2, some knowledge concepts fit well into a strict hierarchy, such as the organization of physical space in the universe. Cognitive radio concepts include cognitive PDAs and their components (e.g. modems, protocol stacks), users, networks, locations, etc. Explicit context is needed to differentiate between concepts with identical syntax but different semantics. "Constellation" in a modem [46], for example, differs from a "constellation" of Iridium satellites [47].

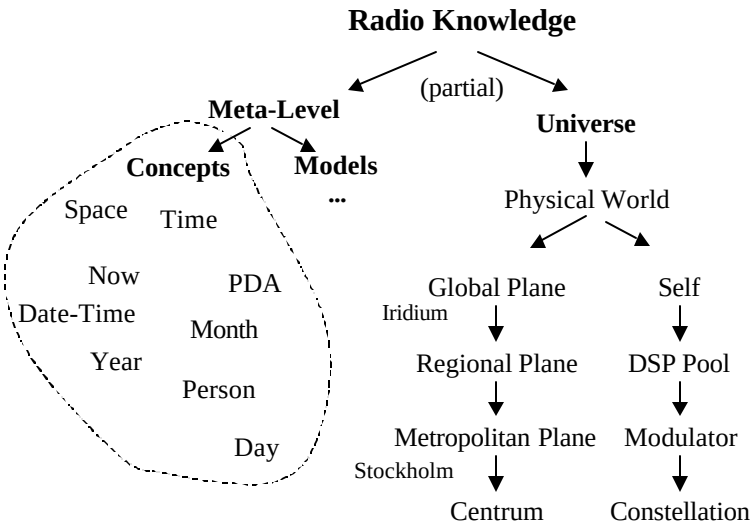


Figure 2-5 A Partial a-priori Radio Knowledge Representation Space

Other more general concepts such as *the existence* of time, space and such abstract notions may all be at the same level of abstraction in meta-space. Meta-space defines what knowledge is present in the more concrete, axiomatic parts of the concept hierarchy. The representation of a-priori concepts in general knowledge-structures facilitates the context-free recognition of concepts as they appear in the environment. It also facilitates the subsequent context-driven interpretation of such stimuli. In addition, the representation of radio concepts in a context-ontology facilitates the incremental acquisition of knowledge via machine learning techniques.

2.4.2 Decision Trees and a Case-Based Approach

In early machine learning research, the focus was on developing new techniques in small-scale domains with simple a-priori knowledge. During the past ten years, the focus has shifted to solving real-world problems such as trouble-shooting industrial machines [48] and defending against computer virus attacks [49]. One branch of machine learning represents concepts in formal logic [50]. The simplest formal logic is the propositional calculus (zero-order logic) where conjunctions of Boolean constants represent concepts. In radio, for example, the following propositional formula might hold:

Message_Coded Ç ***High_SNR*** Þ ***Transmit***

In attribute-value logic, positive and negative examples of concepts may be represented in vectors of the values of attributes. Quinlan [51], for example, used attribute-value logic to

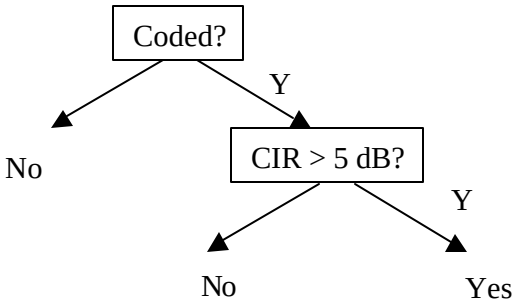
induce decision trees. His algorithm, called ID3 is relevant. If all members of a set have the same value of a given attribute, then that set is a leaf of the tree. Otherwise, select an attribute that partitions the set into orthogonal subsets, and ascribe the value of the attribute to each branch of the tree. This leads to decision trees that efficiently compute rules of the form:

Attribute-Vector(x_0, x_1, …, x_n) Þ Transmit

The associated decision tree shows how examples of a two-element attribute vector of coding and carrier to interference ratio (CIR) might be converted by ID3 into a decision tree for transmission of a message (Figure 2-6).

Coding	**CIR**	**Transmit**
Yes	3 dB	no
Yes	8 dB	yes
No	11 dB	no
No	7 dB	no
Yes	5 dB	yes
No	6 dB	no

Training Set ("Cases")



ID3 Decision Tree

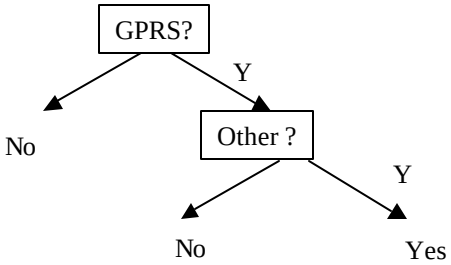
Figure 2-6 ID3 Transforms a Training Set into a Decision Tree

The data vector (Coding, CIR) can be viewed as a situation, a context, or "problem" confronting the radio control system. For a specific status of the convolutional coder and the network (e.g. telling the mobile of its received CIR), should the radio attempt to send a large email file? This kind of decision is not in the traditional purview of the modem. Setting coding parameters as a function of the closed-loop channel characteristics is within the scope of the modem in some HF Automatic Link Establishment (ALE) protocols. But to decide not to transmit, in spite of a CIR that is above the minimum, is a decision usually not taken by a modem. Neither is it in the purview of the traditional protocol stack, which tries to send data if the radio is "ready". However, it would arise in a cognitive radio where the radio is learning from interacting with the environment and the user. The user may provide negative reinforcement for unacceptable end-to-end QoS, excessive time delay, etc. Inference about which RF bands and modes best satisfy needs may take into account long-term average CIR, node movement, time of day, as a function of subjective factors called user communications context. For example, one has different priorities for downloading a map with directions when

shopping near home than when lost and in a storm with the road ahead blocked by a fallen tree.

Time	Service	Context	Transmit
Morning	GPRS	Commuting	yes
Afternoon	GPRS	Commuting	no
Late Night	GPRS	Lost	yes
Morning	3G	Going to Meeting	yes
Morning	GPRS	Commuting	no

Training Set ("Cases")



ID3 Decision Tree?

Figure 2-7 Satisfying a User Given Wireless Availability, QoS and Price

Consider the challenge of machine-learning of radio concepts further. Figure 2-7 suggests how user communications context may influence a decision to transmit on a given RF band and mode. Given only the training set shown, ID3 could yield an inappropriate tree. The cost of 3G could be justified when on the way to an important meeting, but not otherwise. In addition, the training set shows that the user agreed to use GPRS on the morning commute in one case and not in another. This situation provides no information to ID3, but should provide a machine-learning opportunity to a cognitive agent. The agent can find out from the user what specific aspect of the situation caused that decision to be different in the two cases. At least three machine learning approaches apply: conceptual clustering, reinforcement learning, and case-based or memory-based learning. Conceptual clustering [β1] derives predicate-calculus expressions given an unstructured database of such cases. Reinforcement learning extracts rewards from the environment to structure the machine learning. Case-based reasoning (CBR) retrieves and applies cases to new situations.

Case-based reasoning retains sets of "problems" with associated "solutions." This approach is data-intensive because the raw data characterizing the cases are stored. Contrast this to feature-space techniques in which a cluster center and covariance matrix might be retained instead of the 4,000 points from which these parameters were derived. The case-based reasoner would retain the 4,000 data points. It would then attempt to retrieve the most relevant case (data point) to apply the prior solution (e.g. associated class of the data point) to the current problem. Some reasoners revise the solution to better fit the current situation. Successful new solutions are retained with the details of the associated problem for use in the future. This completes the case based reasoning cycle: Retrieve, reuse, revise, and retain. Commercially successful

retrieval processes have been based on nearest-neighbor algorithms, e.g. with roots in statistical pattern recognition. Others have employed decision trees based on ID3 [52]. CBR research systems span the array of machine-learning techniques. If the retrieval process is based on decision trees, then the retention process includes digesting the new cases into updated decision trees. Similar example-integration cycles are needed for alternative schemes (e.g. to generate predicates from conceptual clustering).

Case based reasoning provides a point of departure for machine learning in cognitive radio. A cognitive radio's sensor suite includes time, location, radio network status, and user input. These inputs, plus communications-related inferences derived from them, comprise the generalized user-communications-context. This data structure may be thought of as a vector of attributes of the scene in which a cognitive PDA finds itself. Something like a context vector must be formed and manipulated to generate communications plans and actions that reflect communications context. Thus scene and communications context are closely related.

2.4.3 Rule-Based Aspects

Attribute-value logic may be attached to entities and embedded in expert-system rules in which the knowledge is represented in if-then rules, antecedent-consequent pairs [53]. Rule chaining forward from facts yields conclusions, e.g. in diagnostic applications. Backward-chaining can yield a plan to achieve a goal. In the classic MYCIN [58] diagnostic system, for example, the entities include the patient, sites of infection (e.g. the blood), results of cultures, etc. Attributes of entities were expressed in Entity-Attribute-Value triples, e.g. "The GRAM-STAIN of the CULTURE is NEGATIVE." Here CULTURE is the entity, GRAM-STAIN is the attribute, and NEGATIVE is the value. These supported the rule-based diagnosis of infections of the blood. This approach became popular in expert systems and object-oriented systems developed in the 1980's [54]. An expert system rule has the form:

Rule N: IF(Antecedent) ® THEN (Consequent),

where Antecedent is a conjunction; multiple rules provide disjunctive inference.

In expert systems, rules are maintained in a knowledge base in which any relevant rule may be applied. Some experts systems have clustered rules into Knowledge Sources (KS) so that

only potentially applicable rules are searched [55]. Rule-based expert systems have demonstrated problem-solving capability on hundreds of applications. In the most successful expert systems applications, the human approach to solving the problem is well understood by domain experts and is unlikely to change much over time. Configuration of computer hardware provides a convenient example.

One acknowledged shortcoming of rule-based expert systems is that as the problem set evolves, new rules must be added by a human expert / knowledge engineer, creating the knowledge-engineering bottleneck. In addition, some rule-based expert systems show outstanding performance on the problem set for which designed, but poor performance on problems that are only slightly different. This has been called "brittleness". To overcome brittleness and mitigate the knowledge-engineering bottleneck, Davis' Tieresias system [35] acquired rules from experts by tracing rule schema. For example, when confronted with diagnosis of a bacterial infection of the blood on which the medical doctor and the core diagnostic system, MYCIN, disagreed, Tieresias would elicit knowledge from the doctor. Its interactive protocol could yield new rules. The system treated rules like multi-variable decision trees. It thus represents an interactive version of ID3's batch processing to form decision trees.

What is needed for cognitive radio is a mechanism by which the best aspects of machine-learning, case based reasoning, and rule-based expert systems may be employed in a radio-engineering framework.

2.4.4 The RKRL-CR1 Approach

In cognitive radio, knowledge that could be treated as training data, cases, or rules is structured into embedded models. These are partitioned into micro-worlds to focus reasoning on subsets of the domain. Each chunk of knowledge is part of a stimulus-response model (srModel), a collection of parallel stimulus-response (SR) pairs.

srModel: {Stimulus (i)® Response(i)}, where 0< i <N

The srPairs are parallel (disjunctive) in the sense that when a stimulus is presented, all pairs in the model are evaluated, and the most-applicable model is applied. Model matching accomplishes the retrieval function of case-based reasoning and the rule-binding function of a

forward-chaining expert system. Each srModel is a collection of stimulus-response pairs. The SR pairs are generalizations of rules in that they can represent a case-based example of a problem and a solution. They are generalizations of case-based reasoning in that the solution to one sub-problem can be forward-chained as part of a problem statement of a subsequent problem, behaving like forward-chaining rules. In principle, one could generate multiple hypotheses using the n-best relevant rules, but CR1 as implemented employs only the best match in the spirit of case-based reasoning. The success or lack of success of single-versus-multiple-hypotheses is not a research issue for this thesis. The primary research issue is how these machine learning techniques contribute to cognitive radio architecture, developed below.

In order to meet the isochronism needs of radio, srModels contain only that knowledge relevant to the specific microworld in which the knowledge is embedded. For example, the knowledge about modems is contained in the srModels of the modem micro-world. Knowledge about physical location is consolidated in the Place srModel, etc.

Within each srModel, SR pairs are structured to capture aspects of communications context relevant to the micro-world. For example, a wireless band/mode mode planner could include the following stimulus-response pair (srPair) in its Plan for Carrier to Interference Ratio (CIR):

CIRPlan: Self@Now@Here@CIR, <, 5 ® "Delay Big File" (1)

The left side of this entry into the CIR Plan model is the communications context which specifies to whom, what, when, and where the inference applies. In addition, the stimulus-response pair is treated as a case, an example of a plan that has been used in the past. That is, all srPair in the CIR Plan srModel have the same structure of attributes and outcomes. They may be treated as a database of mutually compatible problem-solution sets. The a-priori structure of radio knowledge permits one to constrain the data structures in each srModel without detracting from the system's ability to detect use context and control the software radio. Complex inferences accomplished by linking multiple srModels and reasoning across multiple micro-worlds.

When srPairs are bound to local context, costs and values accrue. For example, the CIR Plan srModel could include the srPair:

CIRPlan: User@Monday@Home@CIR, <, 5 ® "Send Big File" (2)

This srPair is representationally compatible with srPair (1) as a case. It also competes with srPair (1) for applicability to the current situation. Suppose the user attempts to send a big file on Tuesday at home. The context **User@Tuesday@Home** represents this situation. It binds to the second plan with Monday = Tuesday. Such a binding accrues value in proportion to the match of User and Home to this context. It also accrues a binding cost for setting **Monday = Tuesday**. Although counter-intuitive, these days of the week may be functionally equal as work days when it comes to sending big files. Such bindings set in motion a process of discovering that relationship. The context **User@Tuesday@Home** binds to the first plan with **Self = User**, **Now = Monday** and **Here = Home**. Since there are fewer bindings, the more specific plan supercedes the less specific plan if all binding costs are equal. With differential binding costs, binding Now could cost zero, but binding Here to Home could cost, say 10 units, when the concept Home is currently bound to Work. Differential binding costs may also be a function of the number of times specific stimulus-response pairs have been used with success in the past. For example, if **Self@ Now@ Here@ CIR, <, 5** has been used with success 100 times, but the other plan has been used only once, then the more commonly successful plan might be preferred.

The binding of context to the stimulus of an srModel accomplishes nearest-neighbor retrieval within the CIRPlan model. It also sets the stage for adaptation of this case to the present situation. In CR1, weighted binding is relatively efficient because of the use of hash-maps that limit search. Nevertheless, some srModels lend themselves to decision trees of the type generated by ID3 and its successors. Cognitive radio embeds all of its knowledge into such srModels. By interpreting srModels both as both rules and as examples, cognitive radio incorporates the strengths of both. Expert-system inference capability is thus integrated in a unique way with the automatic knowledge-acquisition capability of case-based reasoning.

In addition, instead of an inference engine that searches a knowledge base to chain rules together, srModels are clustered into micro-worlds defined by the radio architecture of RKRL. This clusters knowledge sources to avoid combinatorial search. The CIR Plan, for example, is part of the Plan Phase that links the measurement of radio parameters to the specific plans for actions that the PDA can take to enhance its delivery of communications services. In the rapid prototype, the linking of these srModels is pre-programmed (using RKRL as a guide), while the stimulus-response pairs are learned. In a more aggressive machine-learning implementation, the

interconnections among srModels also could be learned interactively, e.g. using the Tieresias meta-schema approach. Alternatively, new srModels could be derived from cases, e.g. using ID3 decision trees to add a new model and to restructure the existing internal models.

In CR1, srModels are implemented as Java classes called PDANodes. These nodes loosely emulate aspects of the computational behavior of biological neurons [56]. Some of these are like afferent nodes in the auditory system [57], transforming sensor data into internal data structures and interconnecting results with higher level processing centers. Other nodes have multiple inputs like conventional neural networks, integrating the results from multiple afferent nodes into internal concepts, plans, and decisions. Other nodes are efferent, causing effectors to operate in response to changes in internal state. Conventional neural network weights and sigmoid functions could be realized in these PDA nodes. As cognitive radio evolves, conventional neural-network weighting and learning techniques may be helpful for some aspects of behavior. A simple reinforcement approach is used in CR1 to focus the treatment on the radio aspects versus the machine learning aspects.

2.4.5 Reinforcement

MYCIN introduced a certainty calculus, a mathematical technique for accruing evidence as logic propositions are satisfied [58]. Evidential reasoning is itself a broad, deep technology area that includes Bayesian inference [59], Dempster-Schaffer [60], Fuzzy sets [61], Rough Sets [62] and many related theoretically based and ad-hoc methods [63]. CR1 uses the general form

Reinforce(context, [Stimulus ® Response]).

The stimulus-response pair is positively reinforced when it is used successfully. CR1 acquires every stimulus to which it is exposed. After a period of operations that depends on its memory, CR1 “sleeps” to integrate the new knowledge. When its memory resources become 50% utilized, it begins to discard non-reinforced knowledge. Reinforcement in artificial neutral network framework is based on weights, a sigmoid function, and the error. In CR1, each raw stimulus is reinforced with unit strength. Each stimulus-response pair (or sequence) explicitly reinforced by the user is given an additional additive weight of k in context. Each response receiving negative reinforcement is decremented by k’ in context. When observing new stimuli, CR1 correlates those stimuli, extracting the response that has the best binding, which is in part

based on the quality of the match to the current context and in part on the degree of reinforcement.

Choice(x_i, x_j) = x_i if f(R(x_i), V(x_i), C(x_i)) > f(R(x_j), V(x_j), C(x_j)) and x_j otherwise.

R(x_i) is the degree of prior reinforcement of x_i, V(x_i) is the binding value, and C(x_i) is the binding cost. The function f weights these metrics and maps the result to a decision space like the closed interval [0,1]. The choice of f can be informed by reinforcement learning [64] and complex adaptive systems [65], e.g., the behavior of foraging ants [66].

The management of reinforcement parameters and functions is an important topic for performance tuning of practical cognitive radios in the future. The larger issue is Maguire's question: "How do cognitive radios learn best." This includes both the internal tuning of parameters and the external structuring of the environment to enhance machine learning. Since many aspects of wireless networks are artificial, they may be adjusted to enhance learning. This thesis does not attempt to answer these questions, but it frames them for future research.

2.4.6 Computational Architecture of Machine Learning

Machine learning includes the use of the predicate calculus to extract concepts from data. In such logic formalisms, a concept is a conjunction of clauses, possibly bound to local variables. Horn clauses are the subset of first-order predicate calculus in which clauses (disjunctions) have at most one positive literal [67]. Prolog (for Programming in Logic) is an efficient logic programming language based on Horn clauses. Horn clauses like {Human(x) OR NOT(Greek(x)}; resolve with Greek(Aristotle) to yield the inference Human(Aristotle) [68]. Prolog and expert systems organize knowledge into discrete chunks (Horn clauses or if-then rules) that must be chained in order to perform inferences. Augmenting expressions with computational capabilities introduces Turing-capability into logic programming. This causes the system to be "partial" [22]. For some combinations of inputs, the system may attempt to search for the answer forever. CR1 employs forward chaining, not resolution theorem-proving. In addition, there are limits on the computations performed in an inference. This is intentionally not a Turing-capable framework, but it is total and therefore guaranteed to consume only computational resources whose bounds can be tightly specified in advance. In part, this is to avoid the temporal indeterminacy of unconstrained resolution theorem-proving. In part this is to

avoid the computational indeterminacy of partial functions. Limited-depth Horn-clause resolution may be realized in this computational framework. In radio applications this lack of Turing-computability is an advantage rather than a disadvantage as demonstrated in Appendix B. The processing architecture limits the computational power to that which can be computed in a short, predictable amount of time, and thus also guarantees that a response is delivered in that predicted time for all stimuli.

2.4.7 Automatic Knowledge Acquisition

The srModel is not just another way of embedding a conventional expert system into a radio. CR1 acquires knowledge into existing models without the intervention of a knowledge engineer. In fact, all stimuli are analyzed and retained in "novelty nodes" in working memory. The content of these nodes is integrated into the knowledge base in a sleep cycle that can include the machine learning techniques identified above. During its wake cycle, CR1 recognizes and acquires some types of new information autonomously. Consider the following illustrative Knowledge Query Manipulation Language (KQML) [69] exchange with a network. First, the PDA asks the network to update its internal srModel of the highest available data rate at its present location.

PDA1: **(ask-one :content (? HighDataRate :Here :Now)**

:receiver Network :language RKRL :context WCDMA)

The network responds with an update that is a function of location as follows:

Network: **(tell-one :content (:HighDataRate 2000000 :Place (:Latitude 242200 :Longitude 758833))**

:receiver PDA1 :language RKRL :context WCDMA)

Since RKRL axiomatizes space, the spatial aspect of this dialog has precise set-theoretic meaning. CR1 embeds a Global Positioning Satellite (GPS) sensor, so the (Latitude, Longitude)[3] vector of the response defines the location where the 2 Mbps data rate is available. On the other hand, the network might need to define the boundaries of this high data rate region in terms not

3 This two-dimensional world is used for simplicity in the construction of the prototype. The extension to 2D plus altitude or full 3D is not a research issue of this thesis.

previously programmed into the PDA. Suppose the network response had been:

Network: **(tell-one :content (:HighDataRate 2000000 :Place (:CellSite 322)**

:receiver PDA1 :language RKRL :context WCDMA)

The PDA parses the network's communication and presents the stimulus ":CellSite 322" to its Place srModel. The null response establishes that it has no knowledge of CellSite as a Place (e.g. in its Place model). The PDA needs to acquire new knowledge by extending its srModel for Place. It therefore requests an extension to its Place model by presenting the structure of its Place model's stimuli - (Lat, Lon) - to the network:

PDA1: **(ask-one :content (:Place CellSite 322 :Latitude ? :Longitude ?)**

:receiver Network :language RKRL :context WCDMA)

This query is formed by an algorithm that deals with null responses from models. It checks the input links of its Place model to determine that it is fed by Lat and Lon models. These are the modelNames of the afferent PDA Nodes feeding the Place node. It composes the query to the network to ask it to express its prior response in terms of stimuli that it recognizes. It will adjoin the (Lat-Lon) response to the unfamiliar CellSite 322 expression to acquire an incremental unit of knowledge in its Place srModel. It subsequently parses the network's response, adding (Lat, Lon) $\rightarrow$CellSite 322 to its Place model. The Place, Lat, Lon, and other internal models are structured via the RKRL taxonomy. This provides sufficient context for such focused knowledge acquisition. Although not a general solution to even CR1's machine learning needs, this capability indicates the potential contributions of machine learning to the future of data communications and software radio. The domain of discourse between a mobile agent and today's networks is elementary. That domain can expand using RKRL. As benefits such as enhanced service delivery and more efficient use of spectrum accrue, the domain of discourse will grow. Using the machine learning techniques identified above, it may grow gracefully to embrace the new opportunities. Subsequently it is shown that cognitive radio's integration of machine learning in srModels and micro-worlds will provide a useful mechanism for facilitating the growth of such protocols.

2.5 Natural Language Processing

In the notional dialog between the PDA and the network, the PDA parsed the network's messages. Formal exchanges such as this require minimal tokenization, pattern-matching and binding of pattern variables for a successful exchange. Interactions with users require more extensive parsing of natural language. In fact, the state of the art in natural language processing supports only modest interactions of limited complexity in constrained domains [70]. This section introduces the elements of natural language processing that are relevant to cognitive radio.

Natural language processing includes speech and text processing as illustrated in Figure 2-8. Text processing research includes text retrieval and machine translation. This includes software that translates most grammatical sentences successfully between those natural languages that are in wide use around the world, such as English, German, French, Japanese, and Chinese. Language translation software products address over 700 languages [71]. Translation programs in the past have not done well in interpreting subtleties.

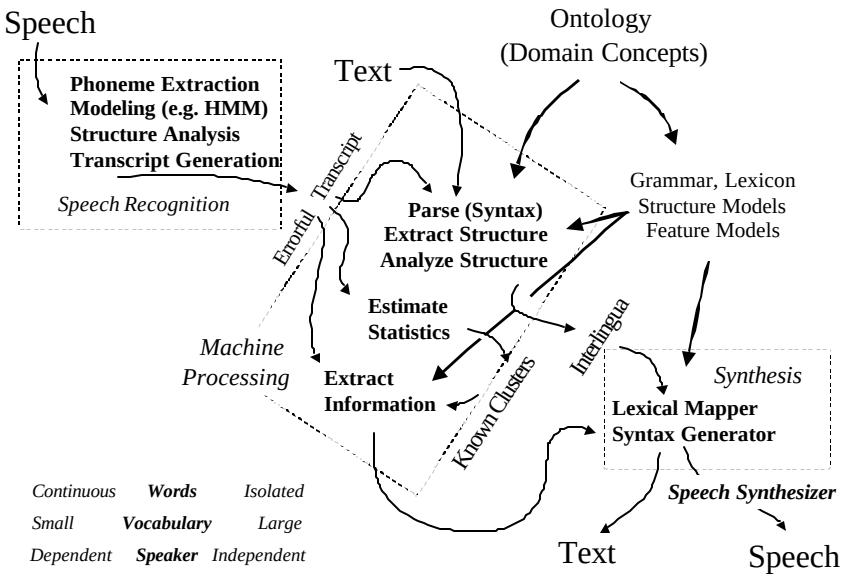


Figure 2-8 Natural Language Processing

They often mis-translated complex or idiomatic expressions. However, they reduced the burden of human translation, e.g. in conducting multi-national business [72], meeting with commercial success. Early progress with the automatic acquisition of rules of syntax [73] helped lay the foundation for substantial progress in the understanding of natural language by computer

[74]. The processing of natural language in text and speech have much in common. The translation of speech from one language to another includes speech recognition [75], machine translation, and speech synthesis [76]. Speech recognition generates an errorful transcript or a hypothesis tree from an acoustic signal, e.g. employing Hidden Markov Models (HMM), Dynamic Time Warping, and/or Fuzzy Sets [77]. The core machine translation step includes parsing of the text against rules of grammar, often embedded into a lexicon. The extraction of semantics from syntax may be accomplished by rules that operate over the resulting data structures. Additional rules resolve anaphora and attempt to deal with other abstractions, including adjusting the semantic model as a function of discourse context [78].

This machine processing flow has much in common with text processing, information extraction (e.g. for data base population from free text) and document retrieval. Performance varies substantially along the dimensions of vocabulary size, articulation of words (isolated words versus continuous speech), and the amount of training (none versus speaker-dependent trained recognition). In addition, noise background and scene complexity (e.g. newsroom versus cocktail party) also determine performance. Word error rates on the order of 40% are state-of-the-art. The best commercial speech processing software has about a 50% word error rate for general untrained speech and about 20-30% for fully trained speaker in low background noise [79]. Fortunately, topics of discourse generally can be detected reliably even in the presence of such high word error rates [80], provided the number of samples is high and the training sets are balanced. Competence in specific domains like airline reservations or telephone directory assistance use a knowledge ontology, an organized set of domain concepts with sufficient expressive power for the task at hand. RKRL provides an ontology for the control of software radios, including interacting with users for radio services and controlling internal states of the radio software.

It is tempting to expect cognitive radio to integrate a natural language processing research system such as SNePS [81], AGFL [82], or XTAG [83] with a morphological analyzer like PCKimmo [84]. These tools both go too far and not far enough in the direction needed for CR1. Since the focus of this dissertation is on radio engineering, it is wise to not be too distracted by the extensive technology of natural language processing. One might like to employ existing tools using a workable interface between the domain of radio engineering and some of the above

natural language tool sets. Unfortunately, the definition of such cross-discipline interfaces is in its infancy. One cannot just express a radio ontology in Interlingua and plug it neatly into XTAG to get a working cognitive radio. The internal data structures that are used in radio mediate the performance of radio tasks (e.g. "transmit a waveform"). The data structures of XTAG, AGFL, etc mediate the conversion of language from one form to another. Thus, XTAG wants to know that transmit is a verb and waveform is a noun. They also need scoping rules for transformations on the linguistic data structures. The way in which domain knowledge is integrated in linguistic structures of these tools tends to obscure the radio engineering aspects.

On the other hand, text retrieval and data mining software such as the Digital Libraries Project has shown an ability to extract documents that share concepts from large text corpora [30]. Thus it seems appropriate to apply the word-vector techniques to detect user context, for example. Unfortunately, the amount of text or speech offered in requests to a PDA is small. For example, a user might say: "PDA, find out about pre-war porcelain from Japan." The user expects the PDA to run an Internet search with the appropriate search terms, to cluster the documents returned, and to provide a useful brief answer. Word-vectors from such brief utterances lack a sufficient number of samples to be statistically meaningful. Thus, the CR1 rapid prototype retains the entire question instead of a derived word vector. If the PDA says "show me" (its default response when confused), and the user enters a query to Yahoo "Porcelain Japan pre-war", the PDA can integrate the question and the action as a stimulus-response pair. The grouping of a "problem" such as a command that the PDA cannot interpret to a "solution" like initiating a Yahoo search forms a case, the point of departure for case-based reasoning. The user's Internet search results could be digested using WebL's document calculus [85], for example.

In addition, cognitive radio relies on contextual cues to behave appropriately. Speaker identification technology is sufficient to reliably identify one speaker from among a small set (e.g. 10 speakers), therefore CR1 assumes that the voices of members of the PDA's household can be differentiated from each other. This is the reason for the SpeakerID sensor of CR1.[4]

Natural language processing systems work well on well-structured speech and text, such as

4 See Figure 4-4 Cognitive Radio Sensor Suite for a complete list of the sensors

the prepared text of a news anchor. But they do not work well yet on noisy, non-grammatical data structures encountered when a user is trying to order a cab in a crowded bar. Thus, other less linguistic or meta-linguistic data structures may be needed to integrate core cognitive radio reasoning with future speech and/or text-processing front ends. Thus RKRL and CR1 employ a limited subset of natural language processing data structures necessary to outline an architecture for the subsequent integration of the technologies. The emphasis of this thesis, then, is on the data communications research required to create a viable cognitive radio architecture, and not on the development engineering required to integrate existing natural language processing tools.

CR1 therefore does not attempt to move the state of the art in speech processing. Hence, speech inputs are simulated using text that would be created by a speech processing front-end with zero word error rate. The expressions are not necessarily grammatical, but they are not very noisy.

2.6 Why Is This Research Important?

This thesis shows how the integration of computational intelligence in radio can benefit the technical operation of wireless networks and the delivery of services to users of emerging 3G systems. To do so, it draws selectively from established bodies of technology outside of radio engineering. These include machine learning and natural language processing. Location-aware computing and communications research has shown that wireless PDAs can benefit human endeavors from travel to the stock market. But the proliferation of radio bands and modes and the increased complexity of the air interfaces, modems, and QoS alternatives creates a need for autonomous assistance to users in the control of such future software radios. A cognitive control system is needed to translate user needs into timely, context sensitive commands to the underlying radio functions of the wireless PDA. This thesis analyzes the need for a cognitive radio both from the perspective of the user and of the wireless network. It then presents a knowledge ontology for radio, RKRL. It explores the contributions of this way of organizing radio knowledge by describing the CR1 rapid prototype. Finally, it summarizes the contributions of the research in a cognitive radio architecture derived from both positive and negative aspects of CR1. After briefly considering previous work in this area, the discussion turns to use-cases, scenarios in which the scope of RKRL and the performance requirements of CR1 are defined.

3 Previous Work

This chapter summarizes research related to the design of cognitive radio and the development of RKRL. Readers who are more interested in the core concepts may safely skip to the next chapter. Those interested in the breadth of research and technical approaches considered in formulating the specific approach to cognitive radio and RKRL described later in this thesis can find that material in this chapter. The first section provides an overview of computational intelligence in telecommunications. The subsequent sections address specific implementation aspects including agent technology, general knowledge representation, control knowledge, radio knowledge, and formal methods.

3.1 Computational Intelligence In Telecommunications

Cognitive radio is an example of agent technology in telecommunications. The compendium, *Intelligent Agents for Telecommunications Applications* [88] and the feature topic on Mobile Software Agents for Telecommunications of the IEEE Communications Magazine [86] summarize the state of the art in applying automated reasoning to telecommunications. Mobility aspects are couched in terms of network applications of "autonomous, interactive, reactive" software objects. Goal-orientation, mobility, planning, reflection, and cooperation are described as additional attributes of agents that distinguish them from other types of software. Although wireless is mentioned in some of the papers, none of the contributions describes anything approaching cognitive radio. Moreover, none of the prior work indicates plans for a radio-specific language like RKRL or even a PDA like CR1. On the other hand, this compendium includes a description of an agent that assists a network operator respond to network faults [87]. It includes a mechanism for learning scenarios and a temporal calculus; these same elements have been considered in the definition of cognitive radio.

Albayrak [88] defines an agent as software that exhibits the functional attributes of autonomy, interactivity, reactivity, goal-orientation, mobility, adaptivity; and that is capable of planning, reflection and cooperation. Driving criteria for agent architectures include scalability, manageability, security, accounting, openness, legacy compatibility, and network residence.

Openness includes the need for agent coordination languages like the Knowledge Query and Manipulation Language (KQML) [89]. But progress reported in the literature has been limited to general principles such as functional integrity [90]. High level architectures address domains other than wireless, such as knowledge of a petro-chemical plant [91]. The agent models that have emerged to date include Aglet, Agent Tcl, Agents for Remote Access (ARA), Concordia, Mole, Odyssey, TACOMA, Voyager, and SHIP-MAI [92]. The focus in the agent literature is on Java and network applications, with minimal attention paid to the physical, data link and radio-related protocols that are critical to wireless. On the other hand, the general discussions of security, portability, mobility, agent communications, resource management, resource discovery, self-identification, control, and data management are generally relevant to the future evolution of cognitive radio.

Busuioc [93] gives lessons learned in the use of agents to synthesize complex services including pro-active offering of services, data download and remote diagnosis of one's automobile. His communications and interaction language lacked an ability to capture that state of procedural knowledge in "a concise form" in order for agents to cooperate. He goes on to say, however, that a central issue is the inability to anticipate functional relationships in advance. While RKRL should include natural language, it must enable model-based reasoning. Such models should provide the bridge between natural language expression of radio etiquettes and computationally explicit implementations. This observation causes one to assess the technology of executable specifications and model-integrated-computing [94]. The relevant literature addresses block-diagram languages for signal processing [95]. These languages focus on the production of executable code when programmed accordingly. The task of RKRL, on the other hand, is to facilitate the machine learning based acquisition of models of the functions and parameters of such code. Thus, this prior work is not as helpful as one might have hoped.

Reilly's comparison of intelligent agent languages also is relevant [96]. He compares KQML, Java, TeleScript, Limbo, Active X and SafeTCL against his criteria for Intelligent Agent Managed Objects. These criteria include target environment, platform independence, execution style (e.g. interpreted versus compiled), native support for agent communications, agent mobility, and security features. Knowledge representation issues were not addressed explicitly. Most of the contributors to this volume [88], however, implicitly modeled user services and

network capabilities. These models were informal, augmented with intuitively accurate definitions, but lacking mathematical foundations in formal logic or ontological knowledge representation. Cognitive radio builds on KQML as a knowledge representation language. Specific "performatives" from KQML are built into RKRL and CR1. In addition RKRL defines explicit model-based semantics for the radio and network applications, facilitating the future rapid evolution of increasingly complex air interfaces, protocols, and applications.

3.2 General Knowledge Representation

Knowledge representation was identified as a core aspect of computational intelligence in the 1970's. Numerous knowledge representation languages (KRLs) have been defined. These include the Common Knowledge Representation Language (CKRL) for the Machine Learning Toolbox [97] and a very general family KL-ONE. SB-ONE [98] (renamed KN-PART+), BACK[99], CLASSIC[100] and KRIS [101] are of the KL-ONE family. SHOE is an extension to HTML in which knowledge ontologies can be defined [102]. MOTEL [103] is a modal KL-ONE implemented in Prolog that contains KRIS as a kernel. COLAB includes a series of representation systems including the logic programming system RELFUN [104] and FORWARD [105]. Telos is a system for temporal and sorted logic [106]. PROLOG represents some kinds of knowledge such as relationships in the predicate calculus. Its limitations have led to the creation of more expressive languages such as URANUS [107]. Uranus extends Prolog using the syntax of Lisp with a multiple world mechanism for knowledge representation and term descriptions. This provides limited functional programming within the framework of logic programming. BinNet is a commercialized Prolog that includes rule-based inference [108]. KL-ONE and its derivatives support research in the representation of general knowledge. These research languages lack domain-specific telecommunications primitives common to SDL.

COLAB/TAXON [109] represents knowledge of terminology within concrete domains. Augmented Transition Networks (ATNs) and case grammars [110] represent knowledge of terminology as well. Lexical Functional Grammars (LFGs) [111] extend the representation of domain terminology to include some linguistic functions of the lexical entities. Although used primarily in the automatic interpretation of natural language text, ATNs and LFGs motivated the local storage of linguistic structure in RKRL's distributed models. COLAB/CONTAX [112] is a

constraint system for weighted constraints over hierarchically structured finite domains. Williams and Cagan describe a qualitative approach to reasoning with such constraints called activity analysis [113]. Their approach applies the well-known Kuhn-Tucker optimality criterion to the arithmetic signs of the cost function and constraint sets to reason over constraint knowledge. Since constraints are such a key aspect of the internal and external environment of radio systems, RKRL incorporates constraint representation.

The most successful knowledge representations related to cognitive radio have been very application-specific. Representation of knowledge about the beat structure of music, for example, required a music-signal-structure ontology. This world view was then "hard-coded" into signal processing algorithms that extract the relevant features [114]. The system included a set of autonomous rule-based agents that reason about the structure of beats in music. The music ontology reported in this research mixed explicit and implicit knowledge structures. That is, the lower layers of the ontology are implicit in performance-oriented code. The knowledge embedded in the code is opaque. RKRL avoids the potential opacity of radio algorithms by extending the ontology to the lowest radio component, data element and executable chunk of code. Because it is defined in XML in a general frame language, RKRL is readily extended to functions and components not explicitly in Version 0.3. This has the advantage of facilitating automatic traversing of the full ontology. It has the disadvantage of being computationally intensive (RKRL 0.3 contains 4,000 frames). Mechanisms for managing computational resources and combinatorial explosion are therefore embedded in RKRL as frame elements that specify the related resources.

Management of a telecommunications network, similarly, requires an entirely different ontology of signal structure, yielding unique syntax and semantics of control in each of several specialty areas [115]. This research in part motivated the inclusion of network-oriented frames in RKRL 0.3.

3.3 Knowledge Query Manipulation Languages

General ontology has been attempted in common sense knowledge (CYC® by Lenat); natural language processing (Generalized Upper Model by Dahlgren); biomedical reactions (GENSIM); ontology definition using first order predicate calculus (Knowledge Interchange

Format, KIF); mechanical properties of ceramic materials (PLINIUS); enterprise modeling (Toronto Virtual Enterprise Project, TOVE); medicine (Unified Medical Language System, UMLS); lexicons (WORDNET); and general ontology (by J. Sowa) [116]. In addition, within a given ontological domain such as control of a radio's air interface, each system relies on an implicit ontology, yet there is apparently no formal (computationally explicit) ontology for radio engineering. RKRL could use the public CYC® ontology to begin to forge a bridge among diverse domains. Such an approach would address the need to incorporate the diversity of domains that impinge on cognitive radio. The CYC® treatment of radio-related concepts mixed technical radio terms with layman's views, making it an inappropriate for RKRL.

KQML [89] represents an agent as a knowledge base with an extensible set of "performatives" expressing a set of beliefs about that knowledge. The literature on KQML embraces a range of topics from agent communications to tools for sharing multiple ontologies. Most of the applications are Internet-driven, however. None address wireless or radio. KQML has been used to represent plans and to exchange plans among agents [117]. It has also been espoused as a suitable language for communications among telecommunications agents [118]. Thus, KQML accommodates axiomatic treatments where ":Content" arguments are Horn clauses, disjunctions or conjunctions. It also supports less formal treatments where ":Content" consists of database queries or natural language. RKRL could extend the domains accessible through KQML to radio engineering.

3.4 Control Knowledge

Control defines limits on system behavior. Classical state-variable control theory encapsulates the behavior of a system in its states and related transition matrices. The next state of a system is a function of its current states and inputs, mediated by a feedback-control law. In cognitive radio, states mediate control as well. Control laws represented in RKRL for cognitive radio include continuous control, such as managing a carrier tracking loop, and discrete control, such as deciding to re-initialize after frame synchronization is lost.

One important historical technique for organizing the symbolic aspects control knowledge is the blackboard system [119]. In a blackboard system, a globally accessible database represents the system's current state of knowledge. Blackboards may express a two-dimensional

database in which one dimension is an inference hierarchy. In signal processing applications, the other dimension typically expresses other independent variables such as space, time, or frequency. Automated reasoning may aggregate lower level data into higher level hypotheses. Knowledge Sources (KSs) map onto and among such levels. Each KS encapsulates just the knowledge required for its assigned set of inferences. Knowledge about the order in which to apply KSs could be maintained in a KS that manages the incoming data. It may also be maintained in ancillary meta-level structures such as a KS that reasons about priority. In the SIAP system [120], a signal understanding system of particular relevance, the blackboard included both the current data and a set of models of objects in the scene. Inference included checking the models for features of the scene and then re-processing lower-level signal data in the scene for finer grain features called out in the model. The author's positive experience with this approach motivated the embedding of a blackboard-like inference architecture of reinforced hierarchical sequences into CR1. Because of the need to parse natural language expressions from users, the blackboard consists of five layers of inference loosely corresponding to character, word, phrase, dialog, and scene. On the other hand, a single inference hierarchy lacks expressive power. Thus, CR1 has a three dimensional inference structure in which the front plane is the Observation plane. Additional planes support the functions of orienting, planning, deciding, and acting. CR1 therefore has multiple inference hierarchies embedding set-theoretic associations among linguistic entities.

Some contemporary systems continue to employ blackboards to represent control knowledge, e.g., in real-time signal processing [121]. KS's that are coded in a procedural language like C are opaque. The knowledge is difficult to access and maintain other than by a programmer who has in-depth understanding of the implicit inter-dependencies among the KS's and the other ad-hoc chunks of code that make the system work properly. KS's written using IF-THEN rules are easier to develop, understand, and debug, particularly if supported by an object-oriented environment with an explanation facility. Even in this case, each KS embeds substantial implicit domain knowledge. This implicit knowledge is thus generally not reusable in other contexts. RKRL therefore encodes knowledge for reuse via an explicit ontology and a mix of knowledge representations from natural language to formal models.

3.5 Uncertainty

Representation of uncertainty has also received much attention in the control literature. Zadeh's definition of fuzzy logic replaces canonical set membership, a (0,1) relationship, with a continuous map from non-membership to membership on the closed real interval [0, 1]. This breakthrough led to many techniques for representing uncertainty [122]. In addition, the Dempster-Schaffer theory of evidence generalized Bayes' notions of a-priori and a-posteriori probability to the more general problem of evidential reasoning [123]. Although theoretically powerful, Dempster-Schaffer requires one to estimate the prior probabilities underlying all possible events. The intractability of this requirement, among other things, has led to a proliferation of ad hoc techniques for representing uncertainty. An observation that underlies the cognitive-radio approach to uncertainty is that probability is a mathematical abstraction of the number of times things occur. This abstraction is intended to allow one to predict those numbers over large populations, such as the probabilities of letters in languages of messages that have been encrypted, which was the original motive for Shannon's famous Theory [124].

Cognitive radio does not impose the probability abstraction on events. Instead, it counts the number of occurrences of events in the environment. It records all stimuli, counting the number of occurrences of each stimulus in context. It prunes out only those that have lowest number of occurrences and only when it lacks sufficient memory for all stimuli. At any point in time, it prefers response A over response B given that Count(S, A)>Count(S, B) for two candidate responses to the same stimulus S in the same context. If there is an a-priori probability distribution of A and B, then the response counts are proportional to the probabilities. There may be no discernable underlying probability space (e.g. when a user makes an arbitrary decision to not want X any more after wanting it for a month). In these cases, the short term count better predicts subsequent events than a long-term probabilistic model. Cognitive radio therefore axiomatizes the counting of events, but does not normalize to probability. Cognitive radio therefore provides a general framework for evidential reasoning. Thus, it supports a mix of Dempster-Sheaffer, Bayes, Shannon, and emerging approaches to evidential reasoning.

For example, rough sets are an emerging research area. They generalize sets using a third truth-value, "uncertain" or "undecideable". Rough sets differ from fuzzy sets in that no degree of membership, probability, or degree of belief in membership is ascribed to an uncertain

member. Therefore, a rough set consists of known members and possible members. A union may either include or not include the possible members, yielding upper- and lower- sets, with corresponding constructs on union and intersection operators. Rough sets are in an early stage of development. Machine learning algorithms have been implemented using rough sets [62]. The approach achieves a degree of generality from the predicate calculus and requires no assignment of degree of membership as with fuzzy sets. This comes at the expense of introducing virtual states and returning uncertain results. RKRL and cognitive radio support rough sets by allowing each KS to interpret the raw data of sets of stimuli and number of occurrences.

The current literature in software agents focuses on computational entities capable of apparently rational behavior. Agent control includes sensing, acting, and planning in a goal-driven way [125]. In one approach, the states of such agents may be mapped using the Belief-Desire-Intention (BDI) paradigm [126]. In this case, there are chance nodes, decision nodes and terminal nodes with corresponding payoff functions. Decision trees are partitioned into those representing alternate worlds. Partitions consist of belief (the reduction of the chance nodes), desirability (the payoff nodes) and intention (the subset of nodes remaining after the application of a strategy such as maximize the minimum expected value). Again, since RKRL does not impose a probabilistic interpretation on uncertainty, a cognitive KS can employ BDI.

To perform consistent logic in uncertain domains requires a calculus that manipulates representations of uncertainty (e.g., numbers) with associated logic (e.g., AND, OR, …). Some powerful uncertainty calculi are non-linear [35]. There is also much relevant technology from probability and statistics literature. Mixture modeling, for example, is the process of representing a statistical distribution in terms of a mixture or weighted sum of other distributions [127]. Cognitive radio will be confronted with uncertainty from sensory systems, and from interpreting user interactions. RKRL therefore embeds facilities for representing reasoning under uncertainty.

3.6 Natural Language Processing

Cognitive radio is to interact with the user via natural language. In addition, CR1 employs natural language processing in the acquisition of structured radio knowledge from RKRL and/or radio engineers. That is, CR1 begins as a machine-learning shell with places for models and

employs machine learning to evolve its personality from training data.

Thus, the entire subject area of natural language processing is potentially relevant to this thesis. This research, however, does not seek to contribute directly to natural language processing technology but merely to employ relevant aspects of established techniques in the architecture of cognitive radio. A few of the best established and most relevant tools are pointed out. The natural language processing tasks required for cognitive radio include the tokenization of text, the extraction of morphological structures, syntax analysis, and the creation of higher level structures that impart meaning. The definition of semantic structures is the focus of this research. These are embedded in the CR1's 3D blackboard of reinforced hierarchical sequences.

Well known tools for lower level text processing include the following. Perl is a script oriented language that can be used to integrate applications on PC platforms [128]. It includes a powerful pattern matcher that can extract tokens from text streams. Tcl [129] has similar capabilities, including a graphics subsystem Tk. PCKimmo [130, 131] performs morphological analysis on the PC. The natural language processing web site [132] includes pointers to additional tools.

Semantic networks, represent knowledge in the interconnections among concepts [133]. The Semantic Network Processing System (SNePS), for example, contains a fully intentional theory of propositional knowledge representation. It includes modules for creating propositional semantic networks, for performing path-based inference, and for node-based inference based on SWM (a relevance logic with quantification). This includes natural deduction. SWM can deal with recursive rules, forward, backward, and bi-directional inference, nonstandard logical connectives and quantifiers, and an assumption-based truth maintenance system for belief revision (SNeBR). Its morphological analyzer (in SNALPS) and generalized ATN parser analyzes and generates natural language. SNACT is a preliminary version of the SNePS Acting component. SNIP 2.2 is an implementation of the SNePS Inference Package that uses rule shadowing and knowledge migration to speed up inference. SNeRE (the SNePS Rational Engine) addresses the integration of inference and acting [134]. Cognitive radio draws on the design and functions of SNePS and SNeRE to define the interface between natural language processing and the core radio-function reasoning. With its relatively broad range of knowledge representation, inference, and analysis capabilities, SnePS motivated the English-like expression

of knowledge as frames in RKRL.

3.7 Radio Knowledge

What domain-specific knowledge should be represented in RKRL? Much of the radio engineer's control of the radio historically has been obtained through the manipulation of the physical device (e.g., changing the value of a capacitor). Electrical engineering languages like SPICE support the design of radio circuits, but much of the radio knowledge represented in the resulting circuits is computationally opaque. That is, a control computer cannot readily manipulate SPICE code describing its circuits to understand its own circuits. It cannot determine how it might change the value of one of its capacitors to enhance reception. It cannot then command the capacitor's value to change and observe the results. RKRL empowers radios to reason about even such entities as capacitors. This is necessary because of the advent of Micro Electro-Mechanical Systems (MEMS) that include capacitors with programmable values [135]. To reason about the control of such values, e.g. to change modes of a software radio, requires reasoning about devices at a higher level. This kind of reasoning is needed for the construction of services using software downloads. RKRL is not intended to facilitate the design of the radio hardware platforms. It does not preclude the inclusion of such knowledge through the extension of the current ontology, but the initial version does not include a SPICE-like capability.

Applications Programmers Interfaces (APIs) are emerging for SDRs [136]. These express the structure of the radio in a computationally accessible format (the API calls, source code and the comments). Unfortunately, much of the knowledge of radio design is implicit in the comments rather than explicit in the language. None the less, such API's are a rich source of information for radio knowledge in RKRL. In addition, Computer aided design (CAD) and simulation environments such as the Signal Processing Workbench (SPW) [137] and Object GEODE [138] describe the internal aspects of radio hardware and/or software in a computationally accessible format. Much of the hardware is now becoming more programmable, such as in Field Programmable Gate Arrays (FPGAs). Radio CAD, simulation, and API packages are thus relevant sources of knowledge for RKRL. RKRL 0.3 includes core hardware concepts into which such detailed knowledge could be grafted in the future. Selected contributions from these sources are included in RKRL 0.3 to represent the radio's internal

structure.

Consider the internal environment of the SDR further. It is clearly necessary to build any control system on a theory of observability, controllability, and stability. Since SDRs are quintessentially computational entities, the control theory must be rooted in computability. Over what domains are radio software functions defined? Does the software require full Turing computability? If so, then the self-extensibility of the radio nodes like CR1 would occur in the context of Turing computability. There are constraints that simplify software extensibility without loss of functional capability for radio applications. In particular, the bounded recursive functions have been shown to be the smallest family of functions necessary for radio and the largest that has the required stability characteristics [139]. In addition, there is geometric structure to the computational interactions among hardware and software and among layers of software. The design of RKRL builds on this work, employing results from point-set topology to add mathematical rigor to the definition of the control structure and its relationship to the hardware and software it controls. If software modules are bounded-recursive then the cognitive radio may draw valid inferences regarding the execution time bounds of modules constructed according to RKRL. RKRL therefore explicitly represents computational resources and related bounds in each of its frames in order to draw inferences about software stability.

3.8 Formal Methods

Formal methods employ an axiomatic treatment of a domain in order to prove theorems about that domain. Formal, model-based semantics is used in natural language processing and formal logic. In this well-known approach, a situation in the real-world is considered a model for statements in the predicate calculus. A valuation function maps aspects of the applications domain into truth-values of predicates. By mapping the results of logic operations back to the domain, inferences describe or predict features of the domain.

Applications include logic programming [140] in which aspects of the domain are represented as facts and axioms. A theorem-prover may prove or disprove hypotheses. In the process, the system may accomplish goals, perform data base queries (or prove that the answer does not lie in the database), and accomplish other useful tasks. In addition, algebraic systems have been developed for the analysis of computing and communications processes. The Algebra

of Communicating Shared Resources (ACSR) and Graphical Communicating Shared Resources (GCSR), for example, are formal languages for the specification, refinement, and analysis of real-time systems [141]. There is also extensive literature on the application of formal methods to wait-free computation. The goal of that research is to design synchronization protocols that are wait free (e.g. any process that is running will finish the protocol in finite time). The conditions under which wait-free protocols can exist are limited [142]. The application of such formal methods to areas directly relevant to cognitive radio includes reasoning about time delays in multiprocessor implementations of software radios.

Other applications of formal models in telecommunications include the formalization of specification and verification models in electronic market contract monitoring [143].

Research on micro-worlds provides some general insights into the formalization of naïve physics knowledge that is readily applied to cognitive radio. Micro-worlds capture interesting representational aspects of a small domain, such as a collection of toy blocks [144]. Recently, Davis described the use of micro-worlds to address competence in the common-sense domain of cutting wood [145]. His research addresses the kinematics of cutting solid rigid objects. The approach is informative. This entails the formalization of the eight separate components illustrated in Figure 3-1. The micro-world itself is the domain about which one is reasoning. In this case, objects occupy time and space; and cut each other, creating new objects. One formal model defines the micro-world in terms of "chunks." In addition, a knowledge base describes the micro-world through the mediation of a language. The domain-specific language is used to define the formal model. In addition, a formal inference system defines a subset of a distinct set of informal inferences that may be drawn about the micro-world. The language relies on an axiomatization, which defines "truth" in the formal model. This axiomatization supports proofs induced by the formal inference system which justify the formal model.

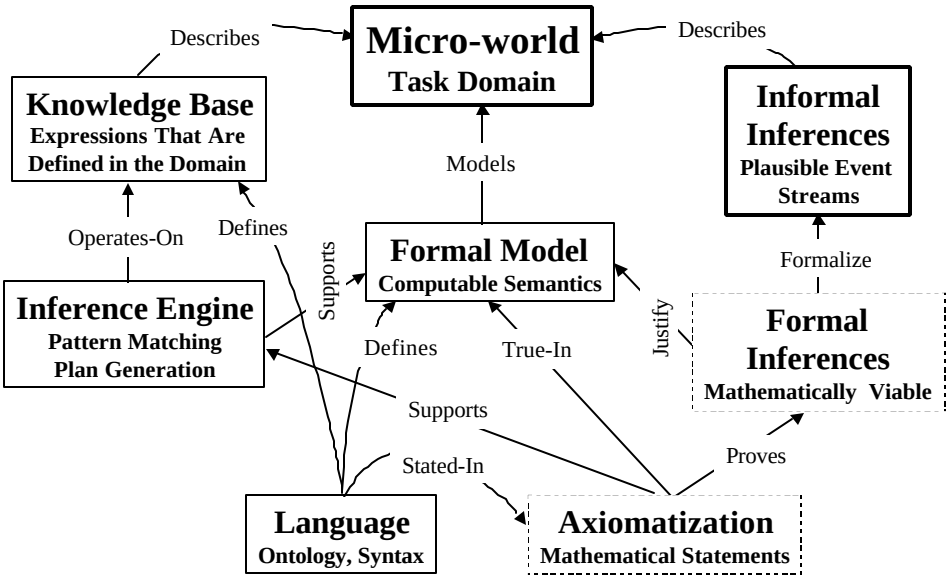


Figure 3-1 Davis' Micro-worlds Framework

Davis' micro-worlds framework is informative. Radio knowledge is a common sense task domain. Cognitive radios should act in a reasonable way in a wide range of circumstances. On the other hand, radio is not a single micro-world. Radio is a large, complex domain in which competence with etiquettes requires a large set of axioms. If structured into a single micro-world, axioms about GSM channels could conflict with axioms about IS-95 channels. This thesis shows how radio can be structured into a set of interacting micro-worlds within each of which one may apply and extend Davis' useful framework.

4 Cognitive Radio

This chapter defines cognitive radio[5]. It begins with an illustration of how capabilities that are missing from current wireless nodes can be embedded in a model-based reasoning framework to enhance the effectiveness of service delivery. In order to extend this concept to include interaction with the environment, a cognition cycle is introduced. This is the top-level control loop for cognitive radio. My licentiate thesis undertook the tedious examination of potential levels of cognitive ability [2], delimiting the bounds of cognitive radio. Cognitive radio's potential abilities require an organization of radio knowledge. In order to address this issue systematically, this chapter identifies four areas of wireless on which cognitive radio could have an impact. These are radio resource management, network management, services delivery, and download certification. Chapter 5 establishes these as design goals for the CR1 rapid prototype. RKRL is defined in Chapter 6. It has the language features that support the cognitive radio performance objectives. This sets the stage for the rapid-prototype implementation of RKRL in CR1 presented in Chapter 7, from which is derived the cognitive radio architecture (Chapter 8).

4.1 Making Radio Self-Aware

Today's digital radios have considerable flexibility, but they have little computational intelligence. For example, the equalizer taps of a GSM SDR reflect the channel impulse response. If the network wants to ask today's handsets "How many distinguishable multipath components are in your location?" two problems arise. First, the network has no standard language with which to pose such a question. Second, the handset has the answer in the structure of its time-domain equalizer taps internally, but it cannot access this information. It has no *computationally accessible* description of its own structure. Thus, it does not "know that it knows." It cannot tell an equalizer from a vocoder. To be termed "cognitive," a radio must be self-aware. It should know a minimum set of basic facts about radio and it should be able to communicate with other entities using that knowledge. For example, it should know that an equalizer's time domain taps reflect the channel impulse response.

5 The term cognitive refers to the mix of declarative and procedural knowledge in a self-aware learning system.

Notionally, then, a cognitive radio would contain a computational model of itself including the equalizer's structure and function. A radio that uses RKRL to accomplish this could be organized as illustrated in Figure 4-1.

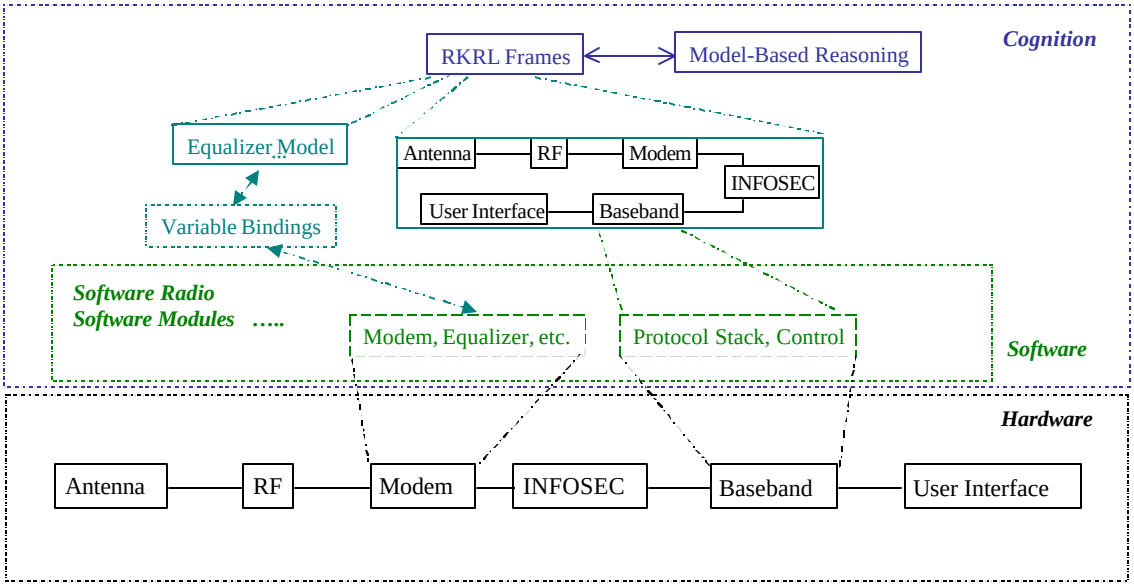


Figure 4-1 Cognitive Radio Framework

The radio hardware consists of a set of modules: antenna, RF section, modem, INFOSEC module, baseband/ protocol processor, and user interface. This could be a software radio, SDR, or PDR. In the figure, the baseband processor hosts the protocol and control software. The modem software includes the modem with equalizer, among other things. In addition, however, a cognitive radio contains an internal model of its own hardware and software structure. The model of the equalizer shown would contain the codified knowledge about equalizers, including how the taps represent the channel impulse response. Variable bindings between the equalizer model and the software equalizer establish the interface between the reasoning capability and the operational software. The model-based reasoning capability that applies these RKRL frames to solve radio control problems gives the radio its "cognitive" ability.[6]

The approach, then, is to represent radio knowledge in RKRL and to structure reasoning algorithms to use that knowledge for the control of software-radios. The radio's model of itself should contain a representation of its functions (e.g. transmission, reception, coding, etc) and

6 The appendix Cognitive Radio: Making Software Radios More Personal considers this example in greater detail.

components (e.g. antenna, RF conversion, DSPs, etc.). In addition, the interfaces among these components should be defined. The SDR Forum has defined these interfaces using the CORBA IDL [13]. This provides a starting point. The Forum has not defined formal semantics that could support machine reasoning about these interfaces, however. RKRL and the cognitive radio architecture derived from CR1 begin to fill this gap. Cognitive radio extends the subjectively grounded definitions of the SDR Forum with formal semantics (e.g. logical sorts and axioms). The CR1 rapid-prototype tightly integrates RKRL frames into the model-based reasoning architecture. The radio's model of its internal structure should then support systems-level interactions with the network. This could include the verbose description of its internal capabilities (e.g. responses of equalizer taps) and the related downloading of a new capability such as an enhanced equalizer.

In addition, however, if the radio is to be context aware, it must interact with the outside world. This is accomplished via the cognition cycle.

4.2 The Cognition Cycle

A cognition cycle by which a cognitive radio may interact with the environment is illustrated in Figure 4-2. Stimuli enter the cognitive radio as interrupts, dispatched to the cognition cycle for a response. Such a cognitive radio continually observes the environment, orients itself, creates plans, decides, and then acts. In addition, machine learning is structured into these phases. Since the assimilation of knowledge by machine learning can be computationally intensive, cognitive radio has sleep and prayer epochs that support machine learning. A sleep epoch is a relatively long period of time (e.g. minutes to hours) during which the radio will not be in use, but has sufficient electrical power for processing. During the sleep epoch, the radio can run machine learning algorithms without detracting from its ability to support its user’s needs. Learning opportunities not resolved in the sleep epoch can be brought to the attention of the user, the host network, or a designer during a prayer epoch.

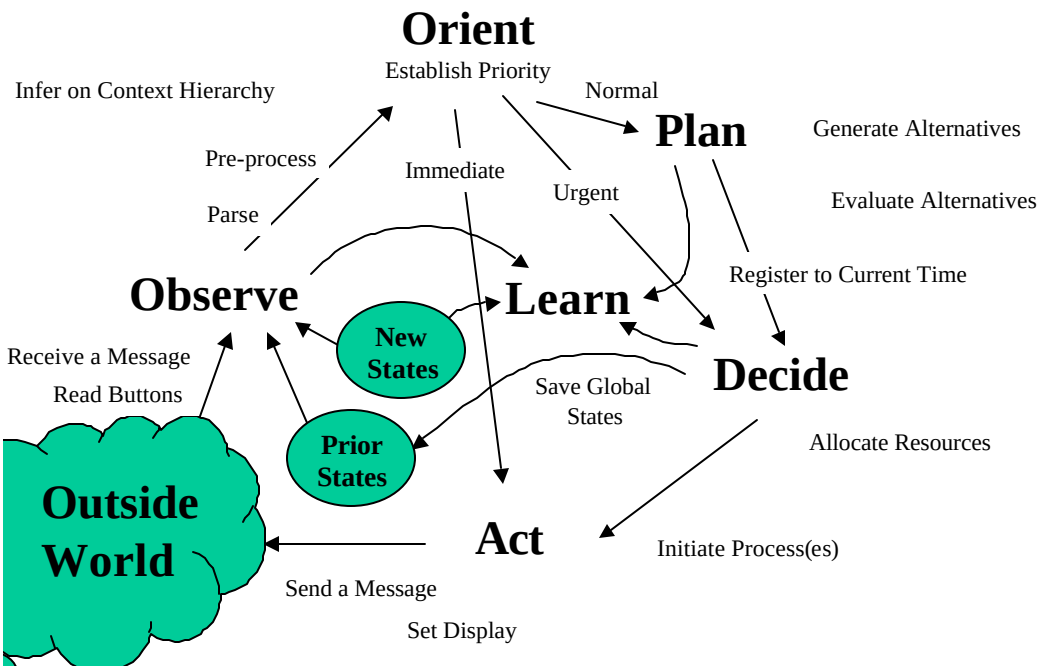


Figure 4-2 Simplified Cognition Cycle

During the wake epoch, the receipt of a new stimulus on any of its sensors initiates a new primary cognition cycle. The cognitive radio observes its environment by parsing incoming information streams. These can include the monitoring of radio broadcasts, e.g. the weather channel, stock ticker tapes, etc. Any RF-LAN or other short-range wireless broadcasts that provide environment awareness information are also parsed. In the observation phase, it also reads location, temperature, and light level sensors, etc. to infer the user's communications context. The cognitive radio orients itself by determining the priority associated with the stimuli. A power failure might directly invoke an act ("Immediate" path in the figure). A non-recoverable loss of signal on a network might invoke reallocation of resources, e.g. from parsing input to searching for alternative RF channels. This is accomplished via the path labeled "Urgent" in the figure. However, an incoming network message would normally be dealt with by generating a plan ("Normal" path). Planning includes plan generation. As formal models of causality [146] are embedded into planning tools, this phase should also include reasoning about causality. The "Decide" phase selects among the candidate plans. The radio might have the choice to alert the user to an incoming message (e.g. behaving like a pager) or to defer the interruption until later (e.g. behaving like a secretary who is screening calls during an important meeting). "Acting" initiates the selected processes using effector modules. Learning is a

function of observations and decisions. For example, prior and current internal states may be compared with expectations to learn about the effectiveness of a communications mode.

The cognition cycle implies a large scope of hard research problems for cognitive radio. Parsing incoming messages requires natural language text processing. Scanning the user's voice channels for content that further defines the communications context requires speech processing. Planning technology offers a wide range of alternatives in temporal calculus [159], constraint-based scheduling [163], task planning [162], causality modeling [146], and the like. Resource allocation includes algebraic methods for wait-free scheduling protocols [142], Open Distributed Processing (ODP) [147], and Parallel Virtual Machines (PVM) [148]. Finally, machine learning remains one of the core challenges in artificial intelligence research [45]. The focus of this cognitive radio research, then, is not on the development of any one of these technologies per se. Rather, it is on the organization of cognition tasks and on the development of cognition data structures needed to integrate contributions from these diverse disciplines for the context-sensitive delivery of wireless services by software radio. Consider first the potential cognition tasks.

4.3 Organization of Cognition Tasks

Cognition tasks that might be performed range in difficulty from the goal-driven choice of RF band, air interface, or protocol to higher-level tasks of planning, learning, and evolving new protocols. Table 4-1 characterizes these tasks in terms of nine levels of capability.

Table 4-1 Characteristics of Radio Cognition Task

Level	Capability	Task Characteristics
0	Pre-programmed	The radio has no model-based reasoning capability
1	Goal-driven	Goal-driven choice of RF band, air interface, and protocol
2	Context Awareness	Infers external communications context (minimum user involvement)
3	Radio Aware	Flexible reasoning about internal and network architectures
4	Capable of Planning	Reasons over goals as a function of time, space, and context
5	Conducts Negotiations	Expresses arguments for plans/ alternatives to user, peers, networks
6	Learns Fluents [149]	Autonomously determines the structure of the environment
7	Adapts Plans	Autonomously modifies plans as learned fluents change
8	Adapts Protocols	Autonomously proposes and negotiates new protocols

The related model-based reasoning techniques are illustrated in Figure 4-3.

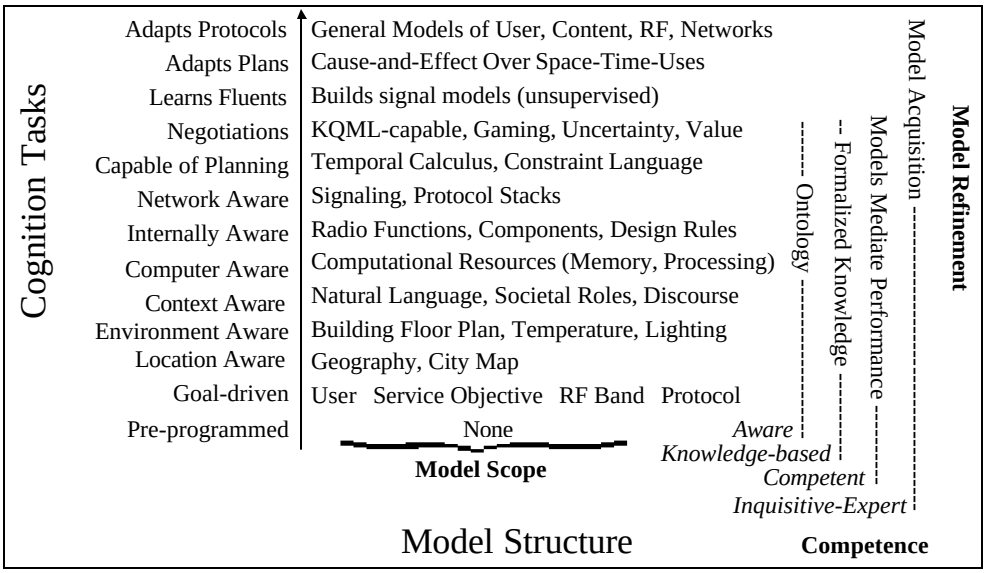


Figure 4-3 Capability Space: Cognition Tasks and Models

4.3.1 The Foundation for Cognition (Capability Levels 0-3)

Cognition capability level 0 represents a conventional PDR, SDR, or software radio. In some cases, the tradeoff between network intelligence and mobile unit intelligence may favor the minimization of computational intelligence in the mobile unit. In the past, the power consumption of hand-held devices was a major design driver limiting processing capacity. During the past five years, however, semiconductor device density has increased 3.5 times while power consumption has been reduced by a factor of 40 [150]. By the year 2002, 0.12 micron production lines may again double device density while the 1.0 Volt power supplies probably will reduce power by another factor of 4. At that point, even handheld devices will support the GFLOP processing capacities and hundreds of Mbytes of random access memory (RAM) needed for the high levels of processing capacity implicit in cognitive radio. The pre-programmed level of cognition may therefore become limited to extremely small wearable devices such as badges supported by cognitive infrastructure [151].

Goal-driven reasoning, cognition capability level 1 in the table, may be achieved with rule-based expert systems technology, and thus is not of research interest by itself. Environment-

awareness extends goal-driven reasoning to goals defined in a spatial context. This level of capability requires a suite of environment sensors and multiband RF like that summarized in Figure 4-4. Approximate location may be known from a global positioning receiver like GPS [152] or Glonass [153]. GPS may not be available inside buildings, and it may not be reliable in urban areas. Therefore, additional positioning sensors are needed. Environments mapped via sensors with wireless read-out or broadcast can provide very detailed data about objects in one's immediate vicinity. Agent-based control of services using the resulting positioning and status information is being explored in a companion research project at KTH [151].

RF Bands and Modes
GSM (IS-136, etc)
GPRS (UWC-136 ...)
3G (W-CDMA …)
RF LAN
AM Broadcast
FM Broadcast
NOAA Weather
Police, Fire, etc.

Environment Sensors
Location:
GPS (Glonass, …)
Accelerometer
Magnetometer (North)
Positioning:
Environment Broadcast
(Doors, Coke Machines, ...)
Timing:
Precision Clock
GPS Clock Updates
Other:
Ambient Light
Digital Image, Video Clip
Temperature

Local Sensors
Speech Recognizer
Speaker ID
Keyboard, Buttons

Effectors
Speech Synthesizer
Text Display
RF Band/ Mode Control

Figure 4-4 Cognitive Radio Sensor Suite

Context awareness (level 2) augments environment awareness in a way that is unique to cognitive radio. This entails the processing of incoming and out-going media in order to infer user communications context. This includes the detection of significant events that may shape the nature of computing and communications services as discussed in the use cases of the next chapter. The processing does not necessarily require in-depth machine understanding of every stimulus. On the contrary, in order to be practicable, all stimuli are scanned for surface features indicating that a context-defining event may be present. If such features warrant, in-depth analysis of the stimulus (e.g. using natural language techniques) may be performed to extract the parameters of the event. Alternatively, the user may be asked what to do in the presence of the surface stimuli. It may then store the situation and user action as a problem-solution case for case-based reasoning. This is the approach taken in CR1, but that does not exclude in-depth

analysis from the resulting cognitive radio architecture.

Level 2 also requires proficiency in air interfaces and protocols (e.g. GSM, DECT, RF-LAN, 3G). One specific objective is to relieve the user from the need to manually select and configure air interface modes as 3G services enter the marketplace. In cognitive radio, this proficiency is obtained by modeling the air interfaces shown in the figure. For every air interface available to the radio, there is a corresponding set of internal models expressed in RKRL. These models describe the mode's behavior, parameters, and bindings by which the cognition cycle can control that mode. The level of detail of control is a function of the richness of the internal knowledge and the level at which the interfaces to those modules have been defined. For example, if the modem is a parameterized software module [154, 155, 156], the cognitive radio could control the parameters of physical layer air interface. If the protocol uses a conventional stack like TCP, then it is likely to be highly encapsulated, limiting the cognitive radio's ability to tailor services. If, however, the protocol is dynamically defined [157] then the intra-module interfaces may be modeled in RKRL and controlled by cognitive radio. If the protocol is defined in a customizable framework, then it may be tailored dynamically to the application [158]. General awareness of the wireline network and its contrasting properties (e.g. short latency, large bandwidths) is also required for radio awareness.

Level 3, radio awareness, in some sense turns the notion of sensing the environment inward. This includes the definition of interfaces between the operational software modules of the SDR and the cognitive control component. In addition, the radio's internal model must reflect substantial knowledge of the behavior of networks. Propagation modeling, QoS models, queuing models, and such system-level models would be characteristic of a cognitive network. A cognitive PDA would include a subset of that knowledge useful in support of local decisions. This could include which of several available RF modes (on different networks) to use given the user's constraints and the PDA's rate of movement and destination, which is a part of communications context for a mobile user.

4.3.2 Core Cognition Capabilities: Natural Language, Planning and Learning

Achieving level 4 capability requires computer-based planning. There is a well-established technology base for computer-based planning [159, 160]. Applications to intelligent control

generally employ some form of hierarchy to modularize the knowledge [161]. A typical decision making architecture from the planning literature analyzes a situation to determine goals. The goals imply candidate plans, each of which is supported or refuted by arguments. Those arguments imply specific plans, which lead to actions, which influence the situation, closing the loop. Domino exemplifies the planning languages that have been published in the literature [162]. Since Domino has classical, temporal, and modal axioms, it represents axioms of beliefs as well as models of time. These approaches tend to ascribe logical structure to situations in which there may be none. For example, a network may change air interfaces or upgrade protocols arbitrarily. A user may change his mind arbitrarily. Thus, cognitive radio incorporates elements of planning technology, but in a way that relaxes the requirement for consistency and completeness in its (prototype) planning processes.

Closely related to planning is level 5, negotiation capability. This is the subject of much current research stimulated by Internet commerce [163]. This includes managing conflicting plans [164]. Conducting negotiations with other radio entities requires an ability to execute negotiation protocols. These may be artificially constrained to finite-state grammars to insure prompt convergence. In addition, some user interactions may be organized as negotiation dialogs. For example, the user may ask for communications services that violate a-priori constraints. The PDA then should express the constraint violations (e.g. “Cannot send this file with a time delay of less than two minutes and cost below five dollars”). It should understand the user’s side of the negotiation (e.g. “Why not?”). It should generate an explanation (e.g. “The file size of two megabytes requires a data rate of over 384 kbps which increases your cost to seven dollars”). Finally, it should understand when the user has made a decision (e.g. “I do not care about cost on this email, so send it”). Thus, effective use of plan-generation capability in interactions with the user may employ negotiation dialogs. This requires substantial natural-language analysis and generation capability along with causality analysis.

Levels 6-8 require machine learning. Statistical learning of patterns is well in hand, but the reliable, incremental acquisition of (valid) new models remains on the frontiers of research. Learning applications in cognitive radio could include autonomously determining the structure of the radio environment as it changes. This would require the learning of fluents [149], events that have duration greater than zero as evidenced, for example, by temporal consistency in a time-

varying stream. An example of a fluent that that a cognitive radio might learn is the fact that the impulse response of a particular set of RF channels is usually not equalized "near the football stadium." A cognitive radio should modify its plans as the fluents change. For example, when the stadium is under renovation, its multipath reflection properties change, so the PDA no longer should us a low data rate when operating "near the stadium." If it always uses the low data rate from prior learning, then it will never discover the availability of the high data rate. This can be solved by advice from the network. But the network would only discover the feasibility of the higher data rate if some small percentage of subscribers attempt that higher data rate.

Ultimately, cognitive radios with these higher capability-levels could propose and negotiate new protocols among themselves. In order to operate in these higher capability levels, radio domain knowledge must be organized for the performance of these cognition tasks.

4.4 Structuring Knowledge for Cognition Tasks

Radio is a physics-oriented domain. Radio propagation, the spatial arrangement of infrastructure, and user mobility are inherently physical phenomena. Thus, naïve physics research offers insights into structuring knowledge for cognition tasks. Davis' micro-world of woodcutting [145] provides a point of departure for the top-level structure of cognitive radio. The micro-worlds framework of RKRL is structured analogously as illustrated in Figure 4-5.

The language, inference engine(s), knowledge base(s), and formal models of cognitive radio are expressed in an integrated language, RKRL. RKRL as a *language* has a set of meta-level micro-worlds that define admissible expressions: inferences, domain knowledge (the knowledge base), computational, and axiomatic models (the formal models). RKRL also has a micro-world for each of its multiple *knowledge bases*, one for each major sub-domain of terrestrial wireless. Finally, RKRL describes *inferences* that can be performed for task-domain automated reasoning. These meso-world components describe and control conventional SDR components (hardware and software).

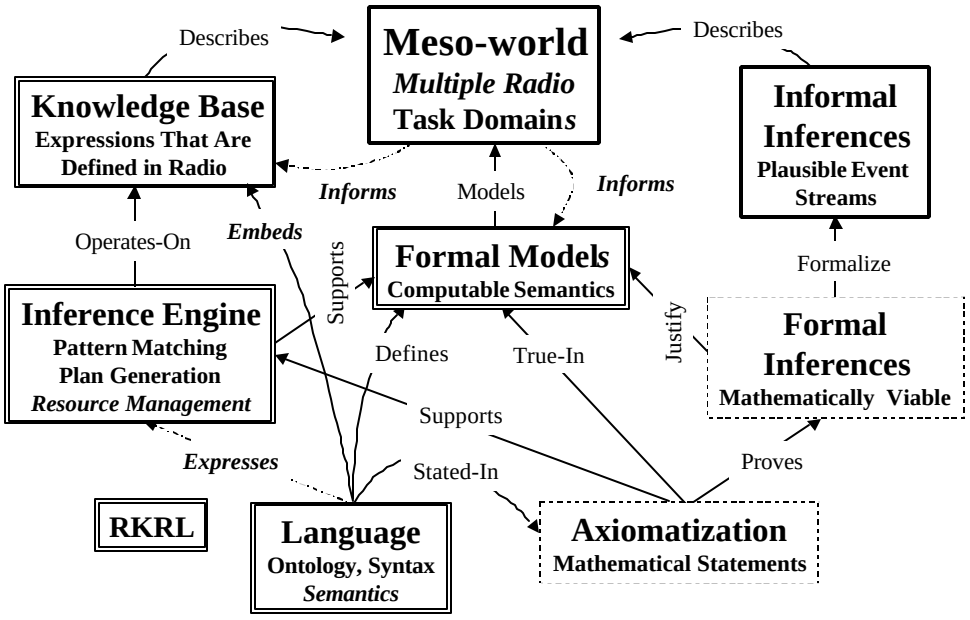


Figure 4-5 The Cognitive Radio Meso-world Framework

The radio task domain is substantially more complex than Davis' woodcutting micro-world. Thus, a single micro-world for all of wireless would suffer from ambiguities, inconsistencies, and combinatorial explosion. Mobile wireless, fixed terrestrial microwave, satellite systems, aeronautical radio and navigation are each complex sub-domains of radio. Each has a different degree of focus on physical phenomena. Satellite communications, for example, includes transionospheric propagation, Keplerian motion, and solar cycles of charging batteries that are not directly relevant to terrestrial wireless. This thesis therefore focuses on the specific radio domain of terrestrial mobile wireless. This domain includes or impinges on military, civil, aeronautical, naval, police, fire, rescue, government, and ISM (Instrumentation, Scientific, and Medical) RF bands among others. Air interfaces include the primary multiple access types: Frequency-, Time-, Code-, and Space-Division Multiple Access (FDMA, TDMA, CDMA, and SDMA). Networks are cellular to some degree because of the physics of propagation, but many are unsupervised (e.g. police and fire). Others are not only supervised by digital control channels (e.g. TETRA[165, 166]), but also highly secure (e.g. GSM [12]. The terrestrial wireless domain also includes users and other entities that must be modeled to infer user communications context. This scope cannot be contained in a single micro-world. Thus, the multiple microworlds of terrestrial mobile wireless constitute a meso-world. The meso-world of RKRL Version 0.3 is partitioned into 47 micro-worlds grouped into the constellations

of Table 4-2.

Table 4-2 RKRL Micro-worlds

Constellation	***Micro-worlds***
Meta Level	Globals, Inference Engines, Meta, Cognition, Goals, KQML, KIF, Skills, Sets, Database, Parsers
Ontological	Universe, Self, Concepts, Models, Time, Space, Spectrum, User
Spatial	Global, Satcom, Regional, Metro, Local
Generic Radio	Architecture, Functions, Internal, Hardware
Software	Software, CORBA, UML, ODP, MPI
Wireless Functions	Air, Modem, Demod, Equalizer, Memory
Protocol Stack	Protocol, Physical, Data Link
Networks	Network, Cellular, Segmentation, Messages, Propagation
Other	References

RKRL constitutes a reference ontology of world knowledge, including concepts, models, radio domain knowledge, etc. The "meta" micro-world defines RKRL's syntax and related meta-level primitives. The semantics are then bootstrapped into the system through the expression of concepts and models. RKRL is built using a set-theoretic operator based on the axiom of choice. That operator is "expand," a structured form of the conventional union operator. RKRL initially consists of the empty set, to which elements are added top-down in an axiomatic way. Elements grow into sets and subsets with finite but otherwise unconstrained structure. The initial set of elements comprises the names ("handles") of the micro-worlds and constellations. These have not changed much in the three iterations of RKRL from version 0.1 to the current version 0.3. Each micro-world may expand to express new knowledge. This raises the following research issues.

1. What minimum set of knowledge must be available in order to approach the performance envisioned for cognitive radio?
2. What minimum set of representation (data) structures is needed to express that knowledge? To expand it as needed? To assure that it may be used effectively in the control of software radios?
3. What is the specific computational relationship between elements of an RKRL and the processes (e.g. of the cognition cycle) that employ that knowledge in cognition tasks?

The approach taken in this research to answering these questions consists of three steps. First, given only the top-level outline of cognitive radio's intended capabilities, one may work through use-cases [167] to determine the software requirements at the next level of detail. The use cases of the next chapter yield an initial answer to question 1. Second, the use-cases also establish representational capabilities from which a minimum set of representational structures may be derived. The results of this analysis answer question 2, as presented in the description of RKRL in Chapter 6. Finally, the statements of RKRL 0.1 to 0.3 were employed in constructing the cognitive radio rapid prototype CR1 in order to derive the answer to question 3. Chapter 7 describes the prototype, while Chapter 8 consolidates the answer to this question into an architecture for cognitive radio. To motivate the use-cases of the next chapter, consider the aspects of terrestrial wireless in which cognitive radio could have an impact.

4.5 Potential Impact Areas

Mobile wireless technology offers voice, email, file transfer, and related services to mobile users. However, fixed wireline networks also offer similar services. In general, the fixed networks have greater bandwidth and availability than wireless networks. On the other hand, the wireless networks provide the services while the user is on the move: commuting, traveling, or just walking around the shopping mall. The vision of third-generation wireless is to provide near wireline availability and high data rates to the mobile user. One anticipated delivery device is the PDA. A cognitive PDA would have the extensive array of sensors summarized above. Cognitive radio may also be embedded in vehicle electronics (e.g. for intelligent highway applications) and in networks. One of the fundamental questions facing the telecommunications industry is the degree to which future mobile PDA's will attract market acceptance versus the desktop/ wireline version of similar services.

Table 4-3 Cognitive Radio Impact Areas

Impact Area	**Contribution**
Radio Resource Management	Etiquette for Spectrum Pooling ("rental")
Network Management	User Profile Protocol
Service Delivery	Service Mediation Protocol (Plan, Execution)
Type Certification	Proof of Stable Behavior

Cognitive radio can have a beneficial impact in those areas listed in Table 4-3. Spectrum pooling, first, is a novel approach to radio resource management enabled by cognitive radio. Pooling is the rental of public and government spectrum by the present owners to cellular service providers (see Appendix D). User profiling, next, extends the present practice of statistically characterizing demand to symbolically characterizing time-space patterns of demand. To do this, a PDA expresses its (user's) plans for movement and use of wireless to the network, reducing uncertainty about future demand. Third, cognitive PDAs can negotiate with the network on behalf of the user to assist users in obtaining needed services. And finally, cognitive PDAs can autonomously manage their internal structure to conform to type certification requirements, assuring stable behavior under program downloads.

These use cases are summarized in Chapter 5. The complete analysis of my licentiate thesis [2] includes simulations that quantify the benefits of cognitive radio. These benefits defined the design requirements for RKRL 0.3 and CR1.

5 Cognitive Radio Use Cases

This chapter analyzes four cognitive radio use cases.

5.1 Radio Resource Management: Spectrum Pooling

Radio resource management includes the assignment of allocated spectrum to communications functions to employ that spectrum efficiently. Spectrum allocation is an institutional process by which governments (and their surrogates such as international organizations) determine the parts of the radio spectrum that are authorized for specific purposes such as communications, navigation, and radar. Resource management and spectrum allocation are related. As technology brings greater economic value to some uses (e.g. cell phones), social pressures mount to re-allocate spectrum from less used bands to over-used bands. Spectrum managers are concerned that social contracts for spectrum uses are fair, meet the objectives of the governments, and are enforceable.

Cognitive radio offers technology-based mechanisms for supporting the social contracts in a way that enhances the utilization of the radio spectrum. One fundamental issue in the application of this technology is the tradeoff between intelligence in the network infrastructure and intelligence in the mobile devices. An assumption of this use case is that there are few limits on the computational capacity of infrastructure. The focus then shifts to the mobile devices. This section seeks to answer the following question:

"What novel contributions can cognitive devices make to the ability of wireless infrastructure to employ radio resources efficiently?"

In general, cognitive radio can apply its awareness of the local radio environment and communications context to enhance efficiency as shown in Table 5-1. This section therefore begins with a parametric analysis of spectrum pooling, a spectrum-rental arrangement that requires radio etiquette offered by cognitive radio that has not been feasible with conventional radio technology. Since the parametric analysis is promising, it includes a statistical analysis of demand offered over space and time. This leads to the identification of specific steps that can be

taken to shape demand in the market transition towards greater use of email and multimedia in wireless. The initial analysis of demand uncertainty motivates the use-case analysis of profiling demand at the source.

Table 5-1 Potential Contributions of Cognitive Radio to Radio Resource Management

Contribution	Mechanism	RKRL Implications
Spectrum Aggregation	Spectrum Rental Etiquette	Structure Modeling
Shape Data Demand	Buffer/Burst Email & Files	Space-Time Awareness
Reduce Demand Uncertainty	Demand Source Profiling	Plan Generation & Matching

5.1.1 Spectrum Pooling

Spectrum pooling, for the purpose of this thesis, is the arrangement under which current owners of spectrum agree to rent it to each other for time periods as brief as one second. Spectrum that could be made available for pooling includes those bands already allocated to mobile terrestrial uses. The large radiation patterns and rapid movement of aircraft and satellites essentially preclude the use of these bands in spectrum pools. In addition, radio navigation and radar bands are not considered in the readily pooled spectrum. This yields relatively large pools summarized in Table 5-2.

Table 5-2 Mobile Spectrum Pools

Band	RF_{min} (MHz)	RF_{max} (MHz)	W_c	Core Applications
Very Low	26.9	399.9	315.21	Long range vehicular traffic
Low	404	960	533.5	Cellular
Mid	1390	2483	930	PCS
High	2483	5900	1068.5	Local, Indoor and RF LANs

The assumptions about radio devices that access these pools are as follows. Within a few years, an affordable software radio PDA should be able to access the *low* and *mid* bands for outdoor use. It also should have a *high* band channel for indoor RF LAN use. Vehicular radios will also be able to access the *very low* band. Infrastructure will access all bands. Thus, any physical cell site at which there is presently a single-band single-mode tower (e.g. a police station) will in the future access all the bands using software radio technology.

The spectrum pooling strategy and related radio etiquette are described in [4], Appendix D of this thesis. This protocol provides for the advertising and rental of spectrum; for the posting of "sold" signs; and for the polite deferral to legacy users. To assure that the spectrum is available to legacy (non-cognitive) radios, renters must employ a protocol that facilitates the near-instantaneous release of the spectrum. Thus, the polite deferral protocol employs 5 ms windows during renting transmissions to receiver listen for legacy interruptions. When these windows occur on 25 ms intervals, a legacy user is assured of immediate release of the spectrum by the renter. This protocol could be essential in fire and rescue bands, for example. The protocol also allows the listen-windows to occur less frequently, increasing the net throughput on the channel. This requires the legacy user to wait longer (e.g. 250 ms) for the channel to be released.

Pooling can be more efficient, if each renting participant (network and mobile) is location-aware. That is, each radio knows where it is located, where the intended receivers are located, and how the emissions will propagate. This knowledge is needed for distributed power management. Pooling efficiency increases further if each radio is environment-aware [26]. Environment-aware radios, for example, know when they are entering a smart building. The building tells them about the availability of a *high band* RF LAN. This knowledge allows them to change quickly to the indoor mode, releasing outdoor radio resources back to the lower bands. Each radio must also be able to differentiate authorized legacy users from other forms of interference. This requires a signal recognition capability. The protocol also provides for an order-wire channel, an ad-hoc signaling channel that can be sustained by the mobile radios for distributed network management. Related protocols are beginning to emerge [168, 169].

5.1.2 Parametric Analysis of Spectrum Pooling

A cognitive radio-access network would employ the etiquette summarized above and described in detail in the appendix. For the parametric analysis, the implication of this etiquette is that the demand of all users may be treated as an aggregate demand against the total spectrum pool. This is not fully representative of implementation constraints, but it permits one to determine the first-order question of whether the concept has potential merit.

With this treatment, a group of cognitive radios and infrastructure could evolve the operating parameters shown in Table 5-3. The total spectrum available for pooling is 1.463 GHz (*mid* and *low* bands). Frequency Domain Duplexing (FDD) is assumed. Voice channels are attributed 8 1/3 kHz of spectrum use in each direction, yielding about 29,270 equivalent voice channels per cell. For simplicity, the entire population of 609,000 individuals offers load to 40 commercial cellular cells and 40 public cells. Offered demand is projected to include multimedia, so it increases from a typical 0.02 Erlang per voice subscriber to a notional 0.1 Erlang in the multimedia future. This yields a total demand of about 761 Erlangs for a net spectrum per user of 320 kHz. Depending on fading and efficiency, data rates range from 160 to 1281 kbps per user on the average. These rates are gross data rates not discounted for Quality of Service (QoS) features such as forward error control which can lower these rates substantially when low bit error rates are required (e.g. for file transfers) [170]. They also do not include signaling overhead. On the other hand, 3G technology is supposed to achieve 0.45 Mbps/MHz/cell [171], so these rates are representative of the ranges achievable with a mix of 2G and 3G technology.

Table 5-3 Illustrative Cognitively Pooled Radio Access Network Parameters

Parameter	**Illustrative Range of Values**	**Remarks**
Total Spectrum	0.4 to 2.5 GHz	1.463 GHz pooled
Duplexing	Frequency Domain (FDD)	Evolved from cellular services
Voice Channel	8 1/3 kHz-equivalent, TDMA or CDMA	Evolved from second generation
Channels per cell	29270	Usable, including 6:1 reuse and FDD
Coverage area	4000 square kilometers	The size of Washington, DC
Number of cells	40 commercial (plus 40 public sites*)	5.5 km average cell radius (3.9 km)
Population	609,000	The entire population of Washington, DC
Offered demand	0.1 Erlang	Multimedia level (vs. 0.02 for voice user)
Demand per cell	761 Erlangs	Increases to 1522 excluding public sites
Spectrum per user	320.4 kHz with public sites)	.16 to 1.28 Mbps per user

* Public cell sites use towers of police, fire, military, and other government/ public facilities pooling spectrum

With pooling, each mobile outdoor user would have an average of 440 kbps. This assumes today's infrastructure density and ~2G-equivalent bandwidth efficiency (0.2 Mbps/MHz/cell). With spectrum pooling, multimedia bandwidths can be achieved without a major increase in the number of cell sites. In part, the sharing of what are now dedicated public sites in other bands increases the number of sites. In addition, the pooling of spectrum is more efficient than block

allocations. Through cognitive radio etiquette, police, fire, and rescue units participating in spectrum pooling will have precedence for spectrum. They can also communicate seamlessly via the shared SDR cell sites.

Appendix D shows parametric variations on the values presented in the table. Figure 5-1 (a) shows the equations of this parametric model. Figure 5-1 (b) shows the values that correspond to the equations of (a). The "Mix" parameter approximates the data rate effects of log-normal fading using a weighted average of the high and low data rates supported by the air interface. The "High Rate" of four bits per Hz corresponds to channel modulation of 16 state Quadrature Amplitude Modulation (16 QAM), a proposed 3G mode. [2] provides further details.

Non-sortable Notional 1G model	
Population	609000
Peak loading	1
Peak	=Peak_loading*Population
Cells	80
Peaking	1
Users	=(Peak/Cells)*Peaking
Offered	0.1
Traffic	=Users*Offered
Wa, kHz	=(Spectrum!C339)*1000
Wc	8.33333333333
Reuse Factor	6
Nc, FDD	=(Wa/(Reuse_Factor))/Wc
VGC/ user	=Nc/Traffic
Capacity, kHz/user	=(Wa/Traffic)/Reuse_Factor
Low Rate	0.5
High Rate	4
High	=High_Rate*Capacity
Low	=Low_Rate*Capacity
Mix (low/high)	3
Service (kbps/user)	=(Mix*Low+High_kbps)/(Mix+1)
Length	50
Width	75
Area	=Width*Length
Cell Area	=Area/Cells
Radius	=SQRT(Cell_Area/3.14159)
Efficiency	=(Service*1000*Traffic)/(Wa*1000)

(a) Equations of the Parametric Model

Non-sortable Notional 1G model	
Population	609000
Peak loading	1
Peak	609000
Cells	80
Peaking	1
Users	7612.5
Offered	0.1
Traffic	761.25
Wa, kHz	1463500
Wc	8.333333333
Reuse Factor	6
Nc, FDD	29270
VGC/ user	38.45
Capacity, kHz	320.4159825
Low Rate	0.5
High Rate	4
High	1281.66393
Low	160.2079912
Mix (low/high	3
Service (kbps/	440.5719759
Length	50
Width	75
Area	3750
Cell Area	46.875
Radius	3.862743652
Efficiency	0.229166667

(b) Corresponding Values

Figure 5-1 Parametric Model of Spectrum Pooling

This parametric analysis addresses the question: "Is spectrum pooling attractive enough in principle to warrant further investigation?" Given the increase in spectrum efficiency suggested by the parametric analysis, the answer is "Yes, it is worth looking into further." So far, however,

the analysis assumes that all kinds of traffic can be merged into a single aggregate demand and distributed wherever necessary in frequency and over space. In addition, it does not differentiate among classes of users who will offer different patterns of demand to the system. A statistical analysis that reflects specific features of cognitive radio and RKRL seems warranted.

5.1.3 A Statistical-Structural Model of Pooling

This use-case examines the behavior of cognitive radio objects in a realistic scenario. Figure 5-2 shows a map of a notional small to moderate sized urban area. Its daily pattern of use of the mobile terrestrial radio spectrum includes demand offered by government entities; police, fire, and other public entities; and consumers. Although commercial services like taxis and delivery vehicles constitute a potentially important distinct niche, for simplicity they were not explicitly modeled in this use-case.

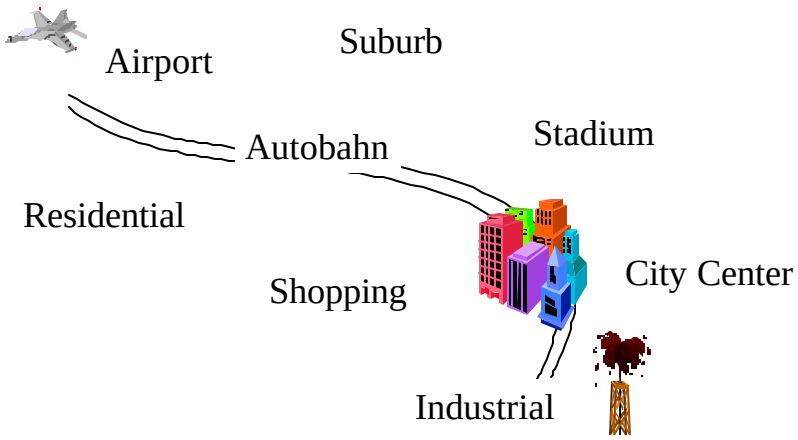


Figure 5-2 Statistical-Structural Use Case City Plan

The daily pattern includes commuting from the suburb and residential area to the city center and industrial area. The airport has a high concentration of business travelers on weekdays and of vacationers on holidays. The stadium area offers relatively low demand except during sporting events. Each of the eight places shown in the figure corresponds to the coverage of one macro-cell site. The top-level structure of the model that was used in the analysis is illustrated in Figure 5-3. The space-time-context distribution allocates classes of user to parts of the city as a function of time of day. The space-time-context model defines the structure of space and time and translates the distributions into fractional allocations of demand to space-time

epochs. The RF Model defines the radio bands, cell sites, channels available to class of user, and tariff structure. The tariff structure is notional but representative so that the relative economic value of the potential cognitive radio contributions can be measured. The user traffic model includes an e-mail and files sub-model that allows one to build multi-media demand from fundamental parameters such as the number of emails a user receives and sends in a day. The Baseline Case model allocates demand to channels and computes total revenue generation and revenue lost due to lack of capacity.

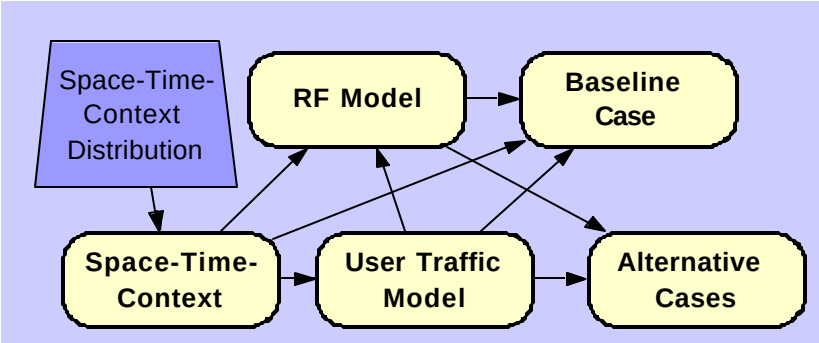
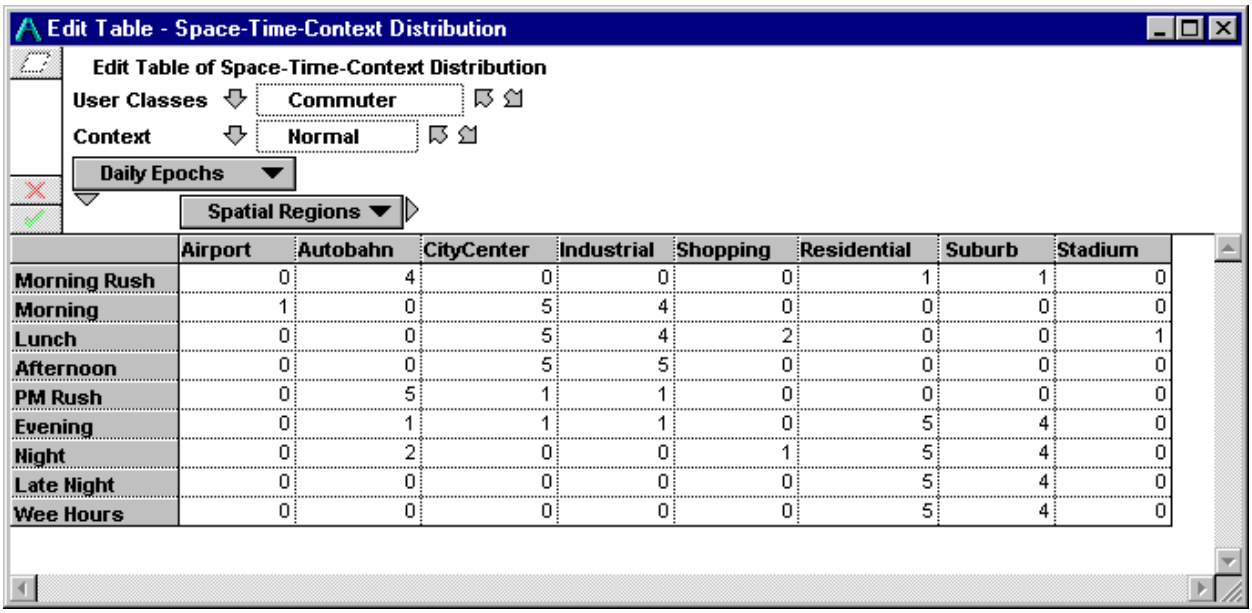


Figure 5-3 Statistical-Structural Model

Users are classified as Infrequent, Commuter, Power Commuter, Police, Fire & Rescue, Government Users, E-mailers, Browser, and Tele-Commuters. The normal day has been partitioned into 9 epochs of 2 or 3 hours each. The spatial distribution of users consists of a uniform distribution over all eight of the spatial regions plus a statistical aggregation of users into specific areas according to the distribution matrix of Figure 5-4.



	Airport	Autobahn	CityCenter	Industrial	Shopping	Residential	Suburb	Stadium
Morning Rush	0	4	0	0	0	1	1	0
Morning	1	0	5	4	0	0	0	0
Lunch	0	0	5	4	2	0	0	1
Afternoon	0	0	5	5	0	0	0	0
PM Rush	0	5	1	1	0	0	0	0
Evening	0	1	1	1	0	5	4	0
Night	0	2	0	0	1	5	4	0
Late Night	0	0	0	0	0	5	4	0
Wee Hours	0	0	0	0	0	5	4	0

Figure 5-4 Space-Time Context Distribution for Normal Commuters

The instantaneous fractional distribution of the user population to locales is given by adding the weights of the matrix to the number of locales to yield a denominator. A matrix consisting of one plus the matrix shown is the numerator. Thus, for example, if the distribution matrix is all zero, the numerator of the fractional distribution is 1 and the denominator is the number of regions (8) for a uniform distribution of 1/8 of the population in each region. The first line shows weights of 4 (autobahn), 1 (residential) and 1 (suburb) indicating that most commuters are on their way to work in the morning rush hour on the autobahn, but some are either still at home or are using the residential roadways. The fractional distributions that result are 7.143% at the airport, 35.71% on the autobahn, 14.29% in the residential and suburb areas and the rest distributed as at the airport. So normally 35% of the commuters are actually in the area of the autobahn, while the rest are at still at home or elsewhere during the morning rush hour.

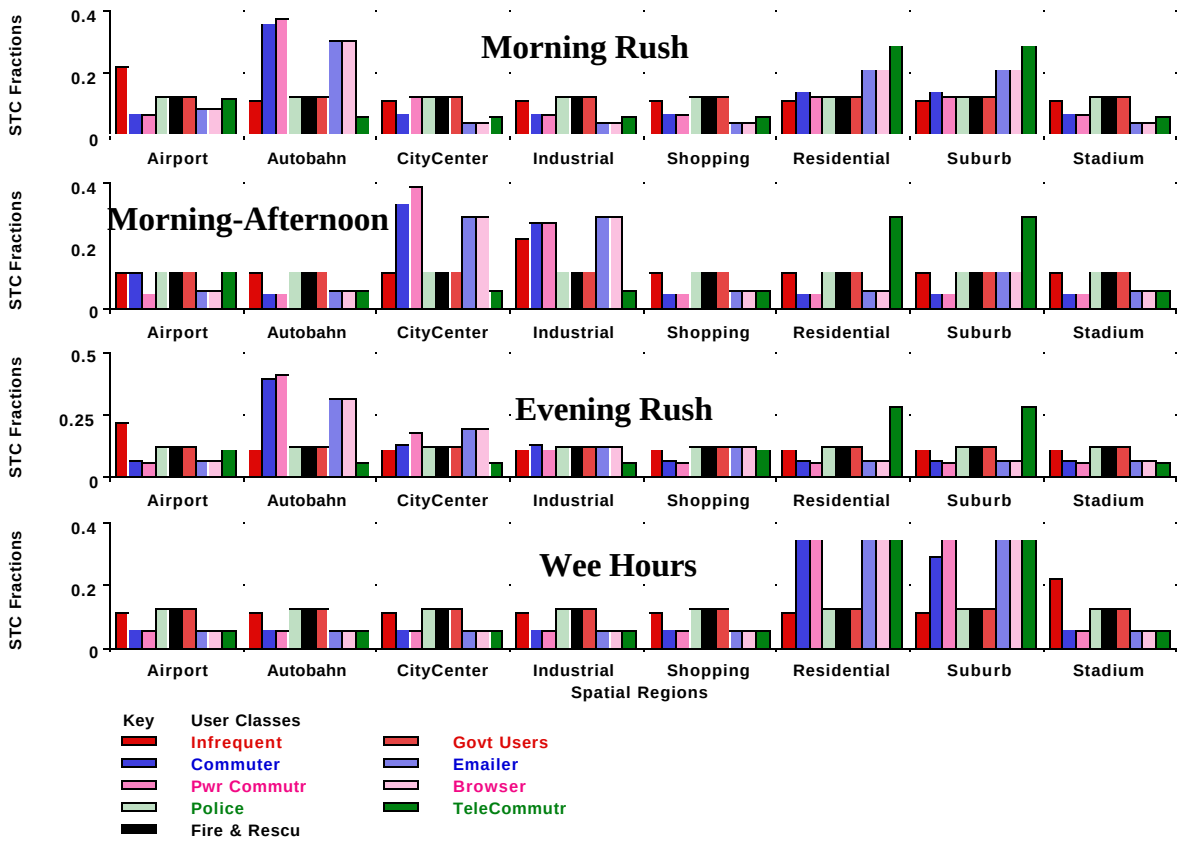


Figure 5-5 Diurnal Pattern of Commuter Locations (partial)

Figure 5-5 shows how commuters move around the city from the suburbs to the city center and back to the residential areas late at night. This space-time aspect of locations modulates the offered demand. Offered demand is defined by class of user and type of content using separate Beta distributions for each period of time. Police, for example, can offer substantial voice traffic during some periods of the night as illustrated in the Beta distribution of Figure 5-6.

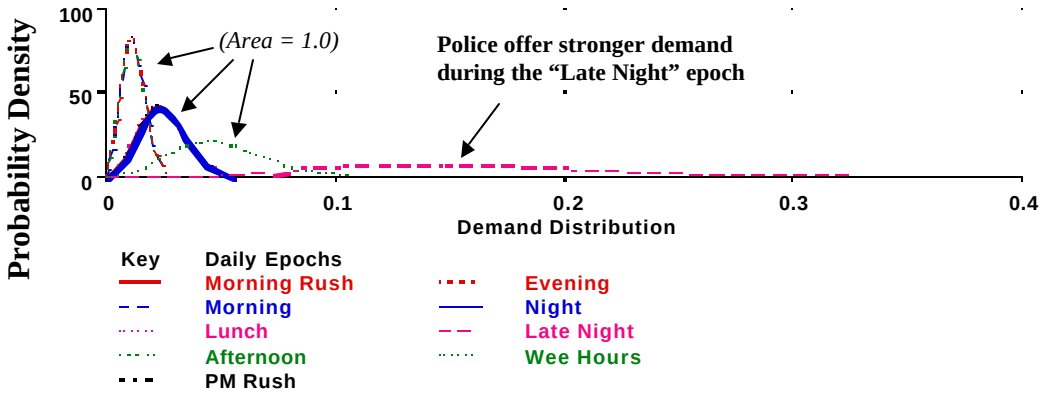


Figure 5-6 Probability Density of Demand Offered by Police

The corresponding cumulative probability density (Figure 5-7) shows that demand is likely to peak at about 0.15 Erlang in the "Late Night" epoch (from 1 until 3 AM).

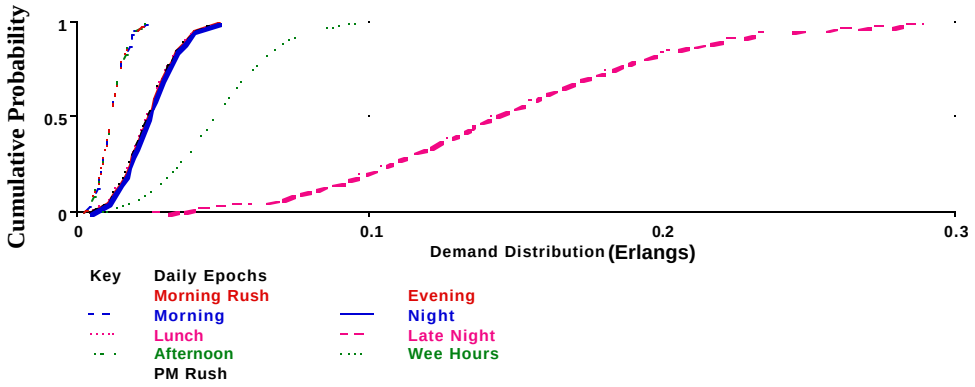


Figure 5-7 Cumulative Probability of Police Demand

These temporal variations of demand are multiplied by the spatial fractions to yield probability distributions of demand as a function of space and time.

The notional wireless infrastructure to which this demand is directed consists of cell sites in each of the eight urban regions. The commercial sector has 100 traffic channels per site in this notional model. The police, fire, and rescue have four sites serving the city center and industrial region; the shopping and stadium; the autobahn and suburbs; and the residential and airport regions respectively. The 100 police, fire and rescue traffic channels are re-used at each of the four sites. The government has one site for the entire urban area, allocating its channels uniformly to demand from any region (this site overlooks the entire area).

A representative tariff structure is illustrated in Figure 5-8. Similar tariff tables exist for pooled and allocated strategies for each of three RF bands (Cellular, Public, and Government).

	Airport	Autobahn	CityCenter	Industrial	Shopping	Residential	Suburb	Stadium
Morning Rush	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
Morning	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2
Lunch	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2
Afternoon	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2
PM Rush	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2
Evening	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
Night	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
Late Night	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
Wee Hours	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1

Figure 5-8 Illustrative Tariffs (Normal Cellular Weekday)

Since there are 167 modules in this Analytica [172] model, a detailed description is beyond

the scope of this analysis. The model includes cross-checking to assure internal consistency of the results. Results of its use are presented in the following section.

5.1.4 Statistical-Structural Analysis of Pooling

The analysis examined the cases shown in Table 5-4. The purpose of this initial analysis is to characterize the contributions of spatial and temporal demand on the effectiveness of the pooling strategy. This serves as both an illustration of the benefits of the cognitive etiquette, and a baseline for the reallocation of demand analysis to come.

Table 5-4 Analysis Cases for Pooling (24 Hour Averages)

Case	**Allocated**		**Pooled**	
	Revenue	**Lost Revenue**	**Revenue**	**Lost Revenue**
Baseline (Mostly 1G)	$136k	$132k	$242k	$47k
Growth with 2G	$145k	$213k	$305k	$87.9k
Explosion with 3G	$147k	$243k	$335k	$109k
Explosion with 1G	$162k	$629k	$458k	$465k

In the baseline case, the population of 37,200 subscribers offers a 24-hour demand of 11,600 Erlangs. In the allocated scheme, about half the potential revenue is lost because of the statistical overload conditions at peak hours. When this demand is spread across 84.6% more channels that become available through pooling, the lost revenue shrinks to only 16% of the total. Note that in addition, the total revenue-generating capability of the system has increased due to the revenue-bearing role of the pooled government and public bands.

The second through fourth cases shows what happens as email and multimedia/ file transfer traffic increases over time. In the fourth case, Explosion with 1G, the infrastructure employs only first generation traffic channels, resulting in huge losses of potential revenue. Although spectrum pooling helps, almost 25% of the potential revenue is lost even with 3G technology. All of this revenue growth is from wireless e-mail, electronic maps, stock broker services on the move, and other services that appear attractive to consumers as mentioned in the background discussion above.

The 2 and 3G cases are interesting. As illustrated in Figure 5-9, different classes of user have different levels of e-mail per day. The scenario employs a weighted average among low, medium, and high e-mail users. In addition, over time, the size of each e-mail file (without attachments) keeps increasing. A migration from 2k Bytes to 200kB was analyzed. In addition, the size and number of e-mail-attachments increases over time (not shown).

	Low	Meduim	High
Infrequent	1	1	1
Commuter	1	1	1
Pwr Commutr	1	1	10
Police	1	10	100
Fire & Rescu	1	5	50
Govt Users	1	10	100
Emailer	1	10	100
Browser	1	10	100
TeleCommutr	10	50	250

Email Technology

Current	2000
Near Term	20K
Far Term	200K

Figure 5-9 Levels of Email Activity and Size (in Bytes) of Each Email Message

At the same time, however, the introduction of 2 and 3G technology will increase capacity per channel as illustrated in Figure 5-10. The baseline case included very little e-mail and multimedia traffic. This traffic is readily handled by narrowband modems with a net rate of only 2.4 kbps (after forward error control and other inefficiencies). The effective data rates for this digital traffic increase as new wireless technology is introduced. The 3G scenario includes a mix of the high rate mobile (384 kbps) and lower data rate bearers as the infrastructure is rolled out to meet the demand.

Data Rates

NB Modem	2400
2G Nominal	8000
GPRS-like	13.4K
3G Low	64K
3G High	384K
RF LAN	7M
Wireline	100M

Fraction Per Scenario

	A Baseline	B 2G	C 3G	D 1G
NB Modem	1	0.7	0.2	1
2G Nominal	0	0.2	0.1	0
GPRS-like	0	0.1	0.1	0
3G Low	0	0	0.3	0
3G High	0	0	0.3	0
RF LAN	0	0	0	0
Wireline	0	0	0	0

Figure 5-10 Channel Characteristics and Distributions per Scenario

The scenario C mix also reflects the fact that not all channels in all bands will be able to support 3G because of the spectrum pooling arrangement. In particular, the 40% of the bands that are using first or second generation traffic channels do so in part because they must give up

those channels to legacy users (e.g. conventional police radios) whose offered demand has not diminished.

The analysis above shows the effect of distributing demand over space, time, and user classes on spectrum pooling. The fact that the etiquette allows all cells to employ all channels is a benefit of cognitive radio. The demand that is offered in these cases is offered exactly when it is generated. Cognitive radio can employ its models of its user, space, time, content (urgency) and communications context to improve the situation further as shown in the next section.

5.1.5 Cognitive Shaping of Demand

Since a cognitive radio maintains a model of the user and the communications context, it has the capability to shape demand. It may offer demand to the network at the point in time (and potentially in space) that balances benefit to the user (e.g. rapid delivery of email, rapid retrieval of items from the web, etc.) against cost to the network (e.g. offering the demand during off-peak periods). Scenario C above (3G in the context of a wireless digital traffic explosion) exhibits the pattern of digital traffic shown in Figure 5-11 (a). In the afternoon, for example, 68% of the demand is either email or file transfers of some type. If only 30% of this traffic is delayed by from one to four hours, there is an increase in total revenue shown in (b). The total revenue potential is decreased because the demand is shifted from peak to off-peak. Revenues increase by about 3 to 4.5 % while total traffic supported increases by about 5%.

Morning Rush	0.367
Morning	0.49
Lunch	0.4076
Afternoon	0.6823
PM Rush	0.2654
Evening	0.3999
Night	0.5211
Late Night	0.5752
Wee Hours	0.5714

(a) Fraction of traffic that is digital

	Baseline Revenue $	Baseline Lost $
A Baseline	242.8K	47.2K
B 2G	305.9K	87.91K
C 3G	335K	109.1K
D 1G	458.6K	465.2K

	Dly Revenue	Dly Lost
A Baseline	252.1K	33.35K
B 2G	317K	75.05K
C 3G	345.6K	97.6K
D 1G	463.4K	472.1K

(b) Revenue Impact

Figure 5-11 Digital Demand Profiles

This relationship between traffic carried and revenue generation is worth pursuing further.

Consider the scenario from the viewpoint of the users. They might not want to maximize the revenue of the network, but rather to maximize their traffic while reducing total cost (their payment to the network). Instead of being merely shifted in time, the digital traffic that is not time sensitive could be diverted to the corporate RF LAN. Assume that only 25% of subscribers are indoors and find the *high band* RF LAN convenient while the other half still require *mid and low band* cellular service. The result is shown in Figure 5-12. Diverting traffic to RF LANs increases total traffic substantially. The upper half of the figure shows the traffic offered without the opportunity to divert to RF LANs, while the lower half shows the value of cognitive delay-shaping as above. In this case, the value of including the RF LAN in the pool is 8.9%, with an additional value of cognitive delay-shaping in the 3G scenario of 4.8% of the total traffic.

Not Pooled

	Lost Erlangs	Real Erlangs
A	13.53K	17.71K
B	21.94K	20.56K
C	25.18K	22.9K
D	69.33K	30.97K

Pooled

	Lost Erlangs	Real Erlangs
A	4553	26.69K
B	8505	33.99K
C	10.63K	37.45K
D	47.85K	52.44K

RF LANs

Pooled

	Lost Erlangs	Real Erlangs
A	2811	28.43K
B	5713	36.78K
C	7305	40.78K
D	36.98K	63.32K

Pooled & Delay Shaped

	Alt Erlangs	Alt Lost
A	29.66K	1318
B	38.4K	4011
C	42.77K	5280
D	65.34K	35.65K

Figure 5-12 Analysis of Offered Traffic Structure (Erlangs)

The conclusion of this analysis is that traffic shaping due to the delay of e-mail and other non-critical file transfers can have a beneficial effect on the overall efficiency of radio resources. In the case investigated, the impact was measurable, but not overwhelming.

5.1.6 Implications for Cognitive Radio

Table 5-5 summarizes the implications of this use-case for the design of CR1.

Table 5-5 Spectrum Pool Use Case Implications for Cognitive Radio

Cognitive Capability	Required Function	Modeling Features of RKRL
Defer to legacy users	Signal recognition	User model to include waveform association
Power management	Location aware	Location taxonomy, propagation modeling
Build order-wire	Protocol structure	Representation of need-structure of protocols
Shape Demand	Smart Buffer/Burst	User Model, Network Load Model, Planning
Source Profiling	Pattern Learning	Space-time, Content, Planning Under Uncertainty

5.2 Network Management Protocols Use Case

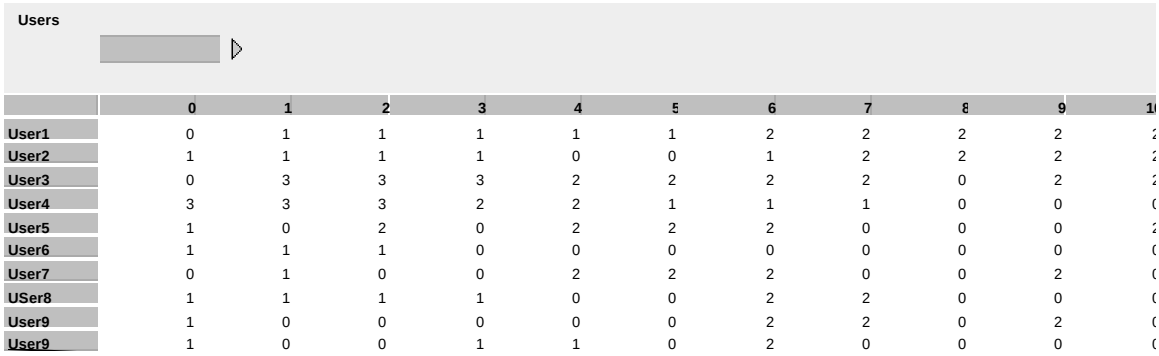
This use case addresses the question of source profiling. Two profiles are considered. The power-user profile is a statistical characterization of a heavy user whose patterns of demand are consistent for some period of time (days, weeks, months, or even years). The second profile includes the registration of an autonomously generated resource-needs plan with the network.

5.2.1 Power User Scenario

This use case is based on an Analytica model of cell handoff strategies. The four cases test the potential value of representing user patterns of demand. In this notional cell structure, the size of each cell is fixed and cells cannot be re-allocated from one cell to another using dynamic channel-allocation techniques. In the first strategy, the baseline, all channels are used for all traffic, both originating in the cell and being handed off from adjacent cells. The second strategy allocates a fixed proportion of the channels for handoff traffic. The third strategy gives priority to "power users" who account for the bulk of the handoff traffic. This strategy requires adjacent cells to reserve space for handoffs based on the presence of a power user in the originating cell. The fourth strategy is based on a plan filed with the network by A COGNITIVE PDA. The plan includes the established pattern of use of wireless including location versus time.

In the following simple data set, the first four users are power users. The table indicates the cell in which the user is located at time instants from 0 to 10. Zero in the table indicates that the user is not consuming traffic capacity, although it may exchange small data messages such as

its own location with the network.



	0	1	2	3	4	5	6	7	8	9	1
User1	0	1	1	1	1	1	2	2	2	2	2
User2	1	1	1	1	0	0	1	2	2	2	2
User3	0	3	3	3	2	2	2	2	0	2	2
User4	3	3	3	2	2	1	1	1	0	0	0
User5	1	0	2	0	2	2	2	0	0	0	2
User6	1	1	1	0	0	0	0	0	0	0	0
User7	0	1	0	0	2	2	2	0	0	2	0
USer8	1	1	1	1	0	0	2	2	0	0	0
User9	1	0	0	0	0	0	2	2	0	2	0
User9	1	0	0	1	1	0	2	0	0	0	0

Figure 5-13 Network Use In Time And Space

The Analytica model of this situation contains 116 nodes. It calculates total offered traffic, call attempt events, forced termination events, revenue-generating traffic serviced, revenue lost due to inability to allocate a traffic channel and other parameters such as cross-checks used to verify the model. The users are offering 64 traffic minutes in eleven minutes (5.18 Erlangs) into a network of three adjacent cells, each of which has only three traffic channels. This heavy use causes 2 forced terminations of users from cell 1 or 3 transitioning to cell 2. It also causes the loss of 17 minutes of random traffic in 13 blocked calls. The assumption is that all channels are available to support all users. Total revenue earned consists of 47 minutes of traffic.

Since forced terminations carry a larger penalty in terms of customer satisfaction than blocked calls, most networks allocate a fraction of traffic channels to handoff traffic. In this simple network, the allocation of only 1 traffic channel comprises 1/3 of the total channels. But the examination of the details of using this strategy is informative. Only 42 revenue minutes are earned with 11 blocked calls and 3 forced terminations. The additional forced terminations occur because of the relatively high density of cell handoff traffic, temporarily exceeding the reserved channels. Since the handoff channels are not available for other traffic, an additional forced termination is caused by user 3 entering cell 2 in minute 4. User 4 occupies the handoff channel from an assignment made in minute 3,. With 5 Erlangs offered in minute 4, what had been a blocked call was converted to a forced termination due to the non-availability of the handoff channel.

In strategy 3, the network differentiates the power users from others based on amount of service used. Since power users have temporal patterns, such as using the car phone during a

long commute home, this service provider decides to force each cell to allocate enough handoff channels to support all power users. This dynamic reservation policy in fact drives forced terminations to zero. However, it also reduces total revenue minutes to 39. This isn't much worse than the 42 earned using the handoff limited policy, but it is low compared to the 47 minutes that the system could support if forced terminations were not an issue.

In strategy 4, the cognitive PDA learns user's fluent of time-space use patterns. It then conveys the plans of the power-user to the network. These plans are not merely statistical. If the cognitive PDA is used as a cell phone to tell a friend of a plan to visit after work, it can let the network know that it does not expect to be on the Autobahn at the usual time. This exchange could be subject to customer agreement regarding the privacy of such information.

The handsets and network track the location and progress of each power user through the cell system. It allocates only 1 minute of blocked channel at the time the power user is about to transition from one cell to the next. Due to random collisions, the algorithm blocks 15 calls for a total of 19 minutes, with 45 revenue minutes earned and no forced terminations. The likely cognitive PDA contribution was simulated through the parameters of the Analytica model. Without a cognitive PDA, this high level of efficiency could not be realized. The cost of a cognitive PDA includes the cost of monitoring the network more closely for plan conformance. In addition, it is possible that a user's behavior would not match the plan perfectly. Many types of observation, planning, prediction and correction of errors could occur. In addition, this example provides a useful case against which to test the CR1 rapid prototype.

The above treatment of channel allocation is simplified. But this simple case establishes circumstances under which the sharing of a plan with a network can reduce forced terminations of high revenue-generating customers. This strategy also improves revenue by 5.3%. The combination of customer loyalty plus increased revenue constitutes a measurable positive impact on radio resource management. The combination of the two could be more significant than the mere statistics of the mathematical analysis suggest.

5.2.2 Planning Use Case Analysis

In strategy 4, THE COGNITIVE PDA learns the plans of the power-users. How should it represent a plan and communicate it to the network? First, the network and the handset need a

means of exchanging knowledge about the user. This could be done in a limited way by programming a fixed set of parameters into the handset, which the network could later read. With today's limited radio platforms, this approach would be acceptable. Cognitive radio could use KQML to represent the request.

(ask-one :content (User ?daily-log) :receiver handset :language RKRL :ontology Time)

Such a stimulus would enter the cognitive radio as an interrupt, dispatched to the RKRL-based cognition cycle. Assume that the present cycle is quiescent, in which resources have been allocated to passively listen to the network. The Observe process would parse the message. If the dialog is to be conducted in expressions that are as close to natural language as possible, must have an ability to interpret the natural language expression "daily-log?" It could recognizes the message header "(ask-one" by matching KQML patterns "(" followed by a performative, "ask-one". This suggests that an ability to automatically form pattern matchers from specifications of message content could be beneficial to the Observe process. Additional details are provided in the companion Licentiate thesis.

5.3 Services Delivery Use Case

Next generation services require support of the information concepts illustrated in Figure 5-14. Wireless has the potential of generating information with a unique degree of focus on the user. This use case addresses the capabilities that cognitive radio needs in order to provide enhanced environment-aware services. An initial baseline scenario shows how a mere location-aware software radio can fail to deliver the required services because it lacks awareness of interactions among radio, environment, user, and services. The cognitive radio scenario then shows how the baseline may be enhanced in order to achieve the required level of performance. The analysis that follows then identifies the features of RKRL needed to represent the radio services domain well enough to realize the high performance cognitive scenario below.

Information Needs
- **Locate people, places, events, services**
- **Understand customs, laws, language**
- **Get help (medical, police, financial, legal…)**

Location Relationships
- **Near curb <-> Need a Taxi?**
- **Toy Department <-> Beanie Babies**
- **Entering Hotel (first time) <-> Reservations**
- **Train Station <-> Ticket, map, directions**
- **Traffic jam <-> Alternate route, e-mail, good book?**

Figure 5-14 Radio Domain Services Representations

5.3.1 Baseline Use Case

Consider a location aware PDA in the use- case scenario suggested in Figure 5-15. The companion Licentiate thesis provides the details of a negative scenario in which advanced services cannot be delivered effectively due to propagation artifacts, ineffective equalization, vocoder inefficiencies, and other common wireless impairments.

- Location-aware GSM "cell phone" info appliance
- "Taxi to Grev Turgatan 16?"
  - Poor voice quality "Direct-Map is Unavailable"
- "Need cash"
  - Incompatible security: "Direct-cash Unavailable"
- "Need real-time language translation"
  - Unequalized multipath: "Direct-Swedish Unavailable"
- Got lost: "Where is nearest taxi stand?"
  - Batteries are dead - I want my money back!!
- Location awareness is a good start, ***but ...***

Figure 5-15 Scenario With Location Aware Cell Phone Type Information Appliance

5.3.2 Cognitive Radio Use Case

The cognitive wireless PDA and information infrastructure contemplated in this aspect of the use case enhances the consumer's experience. Consider the same scenario as above, but with cognitive radio employing the capabilities suggested in Figure 5-16.

- "Taxi to Grev Turgatan 16"
  - Upload Hindi vocoder from handset to network
  - Excellent voice quality results in map retrieval
- "Need cash"
  - Network recognizes incompatible security modes
  - Network asks handset for 1 kbyte for new algorithm
  - Prompts user for date of birth, downloads algorithm
  - "Direct cash available at restaurant around the corner"
- "Need real-time language translation"
  - Handset prompts user to move near the shop doorway.
  - Destructive multipath is now constructive interference
  - Clear real-time translation of Swedish and English

Figure 5-16 Scenario 2 With Cognitive radio Mobiles and Infrastructure

The measures listed in the figure are discussed in detail in the companion Licentiate thesis. The conclusion is that RKRL should represent knowledge of networks, users, user activities (e.g. making a purchase), physical locations (e.g. on a city map), and related radio propagation.

5.3.3 Global Travel Use Case

The act of making a long distance telephone call can be tedious when traveling in a foreign country. In the above scenario, the cognitive PDA simplified the process with a "script from the network". Consider the process by which such a script is created.

Before travel, an agent (human or software) creates an itinerary that is downloaded to the cognitive PDA. The combination of cognitive radio and infrastructure should be able to reason through the scenario suggested in Figure 5-17. First, the download of the itinerary to a workstation should both result in a transfer to the user's PDA and permission from that user to send the itinerary out to the larger network. The distribution of the itinerary should be a function of both the itinerary and the user's normal travel habits. A global model of the earth ("global plane") is needed for international travel. The figure illustrates the use of a notional airline lounge, the "Red Carpet Club" for international travelers who are members. Assume the traveler typically allows sufficient time before flights for contingencies. Consequently, there is a half-hour before departure in the Red Carpet Club. The PDA could log on. If the itinerary includes travel to London, the network could create a script for the PDA to access the home server from

London. Support of this sequence requires both reasoning on the spatial hierarchy, and planning on that hierarchy so that the data needed by the user is available when and where needed.

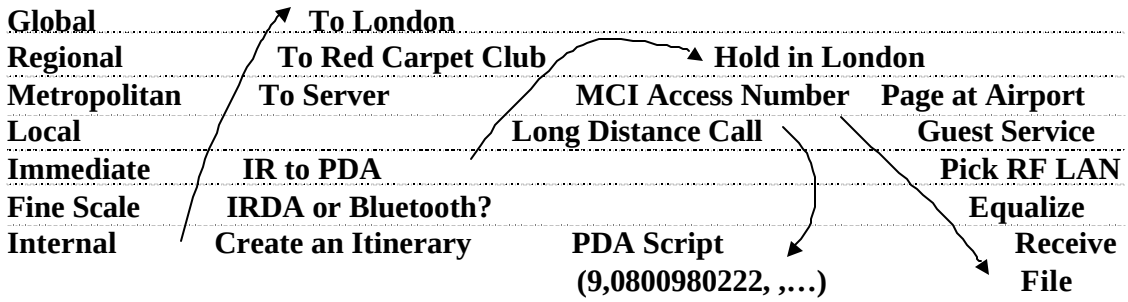


Figure 5-17 Global Reasoning on the Protocol Stack

The PDA has to know that it needs a special number to access a US long distance carrier (say, MCI) from London. This access fact could be associated with a Metropolitan plane on the network. But the PDA has to know to send the request for a script to London in order for the script to be available when needed. This requires a planning component that can reason about network access with respect to a travel itinerary. London then creates a script based on the hotel's PBX codes as in the figure: ("9,, 0800890222,, <MCI>. <Service's 800 number>). If a suitable inter-carrier services support protocol existed, the script would be passed to MCI for its number in London, and back to the home service provider for its 800 number. This script could then be held at the Red Carpet Club. Later, the network would sense that the PDA is in the Club but not logged in. Again as shown in the figure, it could end an autonomous page "If you log in now, you can get your remote access script." If the user says ignores the request, the network holds onto the script, keeps track of me and provides the script later in Stockholm. Alternatively, the user may have empowered the PDA to conduct limited business on the user's behalf. It would then log in by RF LAN as the user walks by the Red Carpet Club, downloads the script, and saves it for later use in Stockholm. Without such support, international travelers sometimes have to make numerous attempts to connect to US and to work the bugs out of a login script. This includes setting the right number of commas for wait time, selecting the proper log-in modes, etc.

This scenario raises questions of trading off privacy versus enhanced services. There are also potentially physical security issues associated with sharing the necessary personal information. But given proper safeguards against compromise and disclosure, one might be

willing to share limited personal information in order to obtain personalized services. Filing a travel itinerary with a cognitive PDA and network might be worth the potential invasion of privacy if its use resulted in truly enhanced services.

5.3.4 Local Reasoning and the Protocol Stack

Support of the cognitive radio services delivery scenario requires reasoning on the ISO protocol stack. The physical layer contributes knowledge of local CIR, fading, and congestion of the wireless network. Competence on this level requires an ability to scan the radio spectrum for activity and to infer the significance for the delivery of wireless services. For example, although a channel may be fading, it could take 20 seconds, say, on average, to register in a different band. Thus, the radio must have some way of knowing (from the network or other mobiles) if the fading will change shortly. It could predict that information (e.g. a propagation model). Otherwise, the decision to change bands and service providers would be uninformed.

The data link layer processing is the domain of a standard software package (e.g. GSM or 3G). But the cognitive aspect of the radio should monitor the behavior of the data link layer to make link-aware choices for the user. The array of choices emerging includes the many parameters implicit in Table 5-6. These are a function of RF band, air interface, and protocol. Thus, goal-driven behavior requires at least a tabulation of these critical parameters as a function of band and mode.

In some cases, the service provider will not offer all data rates. In other cases, the air interface will include some measure of QoS, plus a mechanism for automatically changing among the available modes. Since the link properties are somewhat asymmetrical, the cognitive radio PDA may have some latitude in telling the network what it wants the network to hear. For example, to attain an extremely low BER on a small amount of critical traffic, a cognitive PDA might need to include an excess CIR margin of, say, 3 dB. Should PDA's do things like this? Such an approach could offload aspects of QoS management to the mobile units. In order to perform such a task, the cognitive PDA requires the ability to reason over the mode metrics of Table 5-6. In addition, the PDA should model the implications of the parameters for both the itself (e.g. lower BER) and for the network (greater congestion).

Table 5-6 Illustrative E-Mail Modes

Bearer	Mode	Data Rate	Other Metric(s)	Remarks
GSM	GPRS CS-1	9.05 kbps	Code Rate ½, 40 BCS	Best protection
GSM	GPRS CS-2	13.4 kbps	Code Rate 2/3, 16 BCS	Punctured (CS-2-4)
GSM	GPRS CS-3	15.6 kbps	Code Rate ¾	
GSM	GPRS CS-4	21.4 kbps	Code Rate 1	Highest throughput
IS-136 HS	Class C	<=384 kbps	Delay <1.5s; BER<10^{-6}	BER after ARQ
IS-136 HS	PCS 6	69.2 kbps	Isochronism	But expensive
IS-136 HS	PCS 5	57.35 kbps	Isochronism	PCS n sequence
IS-136 HS	PCS 2	34.2 kbps	Isochronism	PCS 6,5,…1.
IS-136 HS	PCS 1	22.8 kbps	Or CS-4 packetized	Bearer selected
WCDMA	Low Rates	8, 16, 32 kbps	Code rate	All users
WCDMA	Medium	64, 144, 384	Cell loss rate for ATM video	Pedestrian
WCDMA	High Rates	384, 2048	Continuity of rate	Indoors
AMPS	CDPD	< 5 kbps	FEC Protected, low CIR	40 bytes offered
VHF	V.xx	9.6 kbps	Hybrid ARQ	Notional
UHF	V.xx	28.8 kbps	Hybrid ARQ	Notional
LMR	Proprietary	20 kbps	Hybrid ARQ	Notional
IEEE 802.11	2 Mbps	~500 kbps	300% overhead	40 bytes offered
IEEE 802.11	2 Mbps	~1.84 Mbps	8% overhead	1500 bytes offered
IEEE 802.11	11 Mbps	~860 kbps	1179% overhead	40 bytes offered
IEEE 802.11	11 Mbps	~7.59 Mbps	31% overhead	1500 bytes offered
HiperLAN/1	EY-NPMA	~1.52 Mbps	*CAP 0, highest priority	40 bytes offered
HiperLAN/1	EY-NPMA	~15 Mbps	*Channel Access Priority	1500 bytes offered
HiperLAN/2	W-ATM	~7.4 Mbps	DSC0 delay, FEC	40 bytes offered
HiperLAN/2	W-ATM	~16.67 Mbps	DSCx delay, ARQ/FEC	1500 bytes offered
DECT	Notional	155 kbps avg	Interleaved, FEC	Peak 892.4 kbps

Consider the emerging alternatives for exchanging e-mail implicit in Table 5-6. The GSM, IS-136, WCDMA, and AMPS CDPD modes would be available outdoors where the corresponding cellular infrastructure is available. If the cellular infrastructure is unavailable, the VHF or UHF modes with V-series modems could be used, for example, to forward email using pooled spectrum. The Land Mobile Radio (LMR) mode suggests spot-sale of spare capacity from a land mobile service such as a taxi or trucking service. In urban congestion, the use of such modes could route the traffic away from congested cell sites toward uncongested sites.

When the user nears or enters a building, the RF LAN or WCDMA indoor modes become available. IEEE 802.11 and HiperLAN exemplify modes that may become available as RF LANs proliferate. Similarly, local DECT infrastructure may have no airtime cost. Thus, a cognitive business PDA should have the capability to scan the DECT and/or RF LAN bands for

such low cost infrastructure. It should know how to change bearers to minimize total costs within the required service envelope, if that is the goal.

Figure 5-18 shows inferences that a cognitive PDA should be able to make in support of the cognitive services scenario. This entails reasoning on the protocol stack. Contributions of each layer are highlighted in the figure. Reasoning evolves in three stages in this scenario (again from Figure 5-18). The owner begins the use of the appliance by attempting to obtain directions to the nearest antique dealer <1, in the figure>. A cognitive PDA would use the RKRL-defined syntax to parse the owner's query message so that it can adapt its services to the user's needs. The user's outgoing message might be composed by menu selection to query a Merchants database for Antique Dealers in the Local area. The semantics of a query sequence at an information kiosk should be as unambiguous as the use of a bar-code at a check-out counter. Without extensive standardization of such things (unlikely), cognitive radio requires significant natural language processing capability (hard but more likely). The network knows where the user is located, so it can generate a map of dealers <2>.

In addition, the cognitive radio should parse the message in the relevant contexts. Lacking an information kiosk where the user could interact using queries, the cognitive radio should compose an RKRL query for the network:

(advertise :ontology Merchants :content (:seeks "Antique Dealer"
:constraint Local) :sender User :receiver Yellow-Pages)

This KQML statement represents the inference Seeks(User, Antique-Dealer) shown in the figure. In addition, however, the concept of Merchant needs to be defined. So the radio needs a local context within which to reason about merchandise and currency.

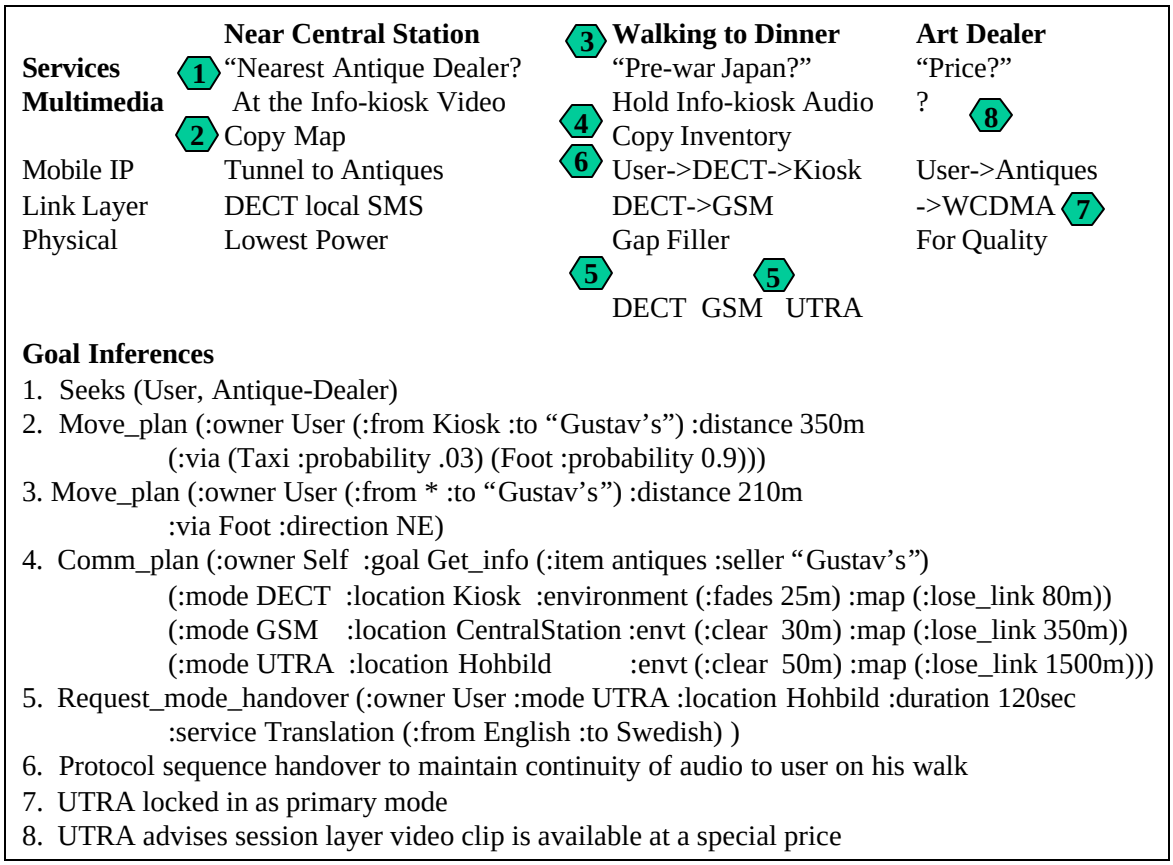


Figure 5-18 Local Reasoning has Protocol Implications

This knowledge could be represented in a notional case grammar (where Object, Plan, and Flow are roles)

(Sell Object Merchandise)

(Sell Flow (Merchandise Merchant->User))

(Pay Object Currency)

(Pay Flow (Currency User->Merchant))

(Buy Plan (Find(#Merchandise)->Check(#Price)->Buy(#Merchandise))

The system must now infer the user's plan to move from the Kiosk to "Gustav's" the antique dealer's shop, which is 350 meters away. The system should also recognize that the user may move by taxi or by foot and that the distances suggest travel by foot as the most likely movement for this user.

In the process of walking to the dealer's location <3>, the user requests further information

using voice queries. First, the act of walking should be sensed locally by the PDA's accelerometers so it is not confused by random GPS or network location perturbations. The sensing of motion at speeds compatible with walking should cause the plan to be updated to a definite movement-by-foot. The user wants porcelain from pre-war Japan. The notional multimode PDA can use DECT, GSM or W-CDMA <4>. Near the kiosk, DECT was in use. But the projected immediate environment includes a fade region 25 meters away and heavy usage that will last for an estimated 30 minutes. GSM is clear and lightly loaded, which could be an unusual condition. And W-CDMA is also lightly loaded. Since only W-CDMA supports video, the system requests a mode change to UTRA/ W-CDMA <5>. Next a protocol sequence is required to continue the tunneling through to the "Antiques Data Base" across the physical layers <6,7>. W-CDMA is locked in as the primary mode. The user asks verbally for the price of specific items <8>. In addition, the W-CDMA offers the user the possibility of a video clip of the porcelain of interest for a promotional price.

In this example, plausible inferences are to be made from data represented in the local and immediate regions. Information could be provided to the user over several mobility environments. The well-informed use appropriate air interface modes should be accomplished by the PDA, in conjunction with the network. In the transition from an era in which wireless connectivity is relatively limited to one in which wireless has a proliferation of bands and modes, the cognitive PDA's have a complex job to do The above scenario suggests how an appropriately constructed inference hierarchy might be employed by cognitive radio to deliver better services.

5.4 Implications – Representing the Services Parameter Space

Goal-driven choice of RF band, air interface, and protocol requires reasoning in at least two dimensions. Source parameters define the information structure of the source bitstream. Service parameters define the tolerance of the source bitstreams to errors and other impairments (Table 5-7). The source parameters have been studied extensively [173]. In addition, Quality of Service (QoS) metrics for the wireline system have been normalized for wireless networks [170].

In principle, a simple goal-driven radio could be outfitted with tables of service-quality parameters as a function of band and mode for the bands and modes it can access. When an

additional air interface is downloaded to the radio, its service parameters could be downloaded as well. Again, in principle, the goal-driven radio could select the band and mode that is available and that best meets the source requirements.

Table 5-7 Critical Mode Selection Decision Parameters

Information Structure	Parameter	Remarks
Source Bitstream	Bit Rate	Quality and coder complexity
Burstiness	Constant (CBR) or variable (VBR)	Min, max, sustained
Isochronism	None, real-time or near-real-time	Data transfers are not isochronous
Burst parameters	Maximum burst size	Large files create long bursts
Tolerance	Tolerance of parameter mismatch	See service quality parameters
Service Quality	Error rates	Losses and delays
Bit error-related	Bit (BER), symbol (SER) errors	Requires error control
Delay-related	Transfer delay, variance, jitter	Excessive delays result in loss
Buffer-related	Packet or Cell Loss Rate (CLR)	Overflows may preclude desired rate
Reliability	Probability(link), Grade of Service	Link and network provisioning
Assuredness	Authentication, privacy	Significant for e-commerce
Cost	Peak, off-peak, service-related	Can be the critical factor

Wireless decision parameters vary temporally with location, time of day, exact frequency subband, traffic elsewhere in the network, weather, and other contextual parameters. There are daily and weekly patterns of business, sports and leisure that shape demand and hence congestion of bands. They also vary cyclically with month of the year. Cognitive radio can adapt its use of spectrum bands and modes if it has an appropriate set of traffic parameters and space-time patterns. It also must include computational models that represent the implications for dynamic context-driven planning (Table 5-8).

Table 5-8 Contextual Parameters For Planning

Information Structure	Parameter	Remarks
Traffic Structure	Statistics of offered load	From radio resource monitoring
Temporal Cycles	Diurnal, weekly, annual, other	Identify periods of peak demand
Location information	Propagation, services	Services and bandwidth versus location
Global Bands/ Modes	Goals (e.g. cost vs. performance)	Tabulate mix vs. geography
Weather	Precipitation, road conditions	Highway routing, emergencies

Information about traffic structure is used in admission control and handoff. Historically,

the network measures these parameters, but does not share this information with the mobile(s). Cognitive radio may employ parameters provided by the network or infer parameters by scanning the spectrum. As suggested in the table, the traffic structure varies as a function of time and space. A cognitive PDA learns user patterns of offered demand and then treats these as a-priori models of temporal patterns to generate expectations about the radio environment. The reinforcement levels of patterns of offered demand in some sense predict the pattern of future demand, such as the use of the PDA's cell-phone features while commuting. Since a cognitive PDA also monitors communications and email for communications-related events, it can identify user states that would shape the offered demand. This allows the network to infer, for example, the level of multiple access interference is likely to increase or decrease during a given time interval. Global roaming continues to increase. Global travelers constitute a small but critical niche market of customers who value global interoperability and can pay for it. To support such a niche, cognitive radio architecture should support the capability to upload a-priori parameters based on a traveler's itinerary. It also needs to observe, validate, and update the critical parameters. Additional parameters useful in the determination of context are summarized in Table 5-9

Table 5-9 Additional Context Parameters

Information Structure	**Parameters**	**Remarks**
Immediate relative position	In pocket, building, on the ground	Thermal and light sensors
Orientation	Direction of travel	E.g. from video frames
Travel	Mode (Auto, train, air, foot), speed,	Correlation to known routes
User Profile	Identity, use patterns	Current and historical
User speech profile	Topics of discourse, needs	From parsing speech traffic
User data profile	Topics of interest, needs	From parsing data, email
Immediate needs profile	Scope qualifiers	Life/death, business, pleasure...
Linkage needs	Criticality of remaining connected to	Emergency, medical, commercial,
Security profile	Authentication, encryption	Identity of trusted entities
Intended recipient(s)	Generic, class, instance	Unknown, any doctor, my wife

These additional parameters define context to sufficient breadth and depth to allow the cognitive radio to make well-informed choice of band and mode as a function of offered traffic within a well-defined operational context. The immediate relative position, for example, can be sensed thermally (in pocket/ in hand). Intensity of sunlight versus GPS availability and signal strength can be used to infer whether one is inside or outside of a building. Direction and rate of

movement can be inferred using imaging sensors and signal processing. Mode of travel may be inferred through monitoring fade rates, which are different for pedestrian versus vehicular transportation. Each mode of transportation implies constraints on band, mode, and ability to support desired quality of service.

Cognitive radio thus will employ a rich set of user profiles. These include the identity of the user, and the user's historical patterns of use, organized as a function of context. In addition, cognitive radio must have the skills to passively observe the user in the communications environment to infer the communication contexts. The user's speech patterns need to be mined for the presence of context triggers found in the cognitive radio's context taxonomy. These include word patterns related to medical emergencies, business transaction, sports, and other use cases. Email and other textual forms of communication also provide rich sources of contextual cues. In addition, the user may have some specific role in society that imparts communications preference, such as for medical professionals, fire fighters, etc. These aspects are part of extended context that is represented in RKRL and embedded into CR1.

5.5 Type-Certified Downloads

According to the SDR Forum, downloaded software will fall into distinct categories [13]:

- High-level communications and computing *applications*
- *Protocol entities* for modification or changing of the air interface or the bearer service
- Low-level *signal processing algorithms* for modification or changing of the communication physical layer processing

Again, according to the Forum, downloads will enable the following applications:

- Download of new computing and communication applications
- Download of new user interface (look and feel) and I/O drivers
- Adaptation of air interface to implement a new standard (inter-standard adaptation)
- Adaptation of air interface to implement different features (e.g., increased bearer data rate) specified within a standard (intra-standard adaptation)
- Download of incremental enhancements (module or entity replacement)
- Download of patches for software bug-fixes
- Download of reference material, e.g, locally available services and operators

- Download of activation licenses to activate downloaded applications, upon verified receipt of payment

5.5.1 Download Scenarios

The following methods of downloading software are envisioned by the Forum:

- Distribution of new software via *SIM-card or other removable media*
- Downloading software via a modem and fixed network, e.g., telephone or cable service
- From a handheld field device
- From CD-ROM or Internet/Intranet, via a PC;
- From a street side terminal, e.g., download onto SIM card via an Automatic Teller Machine (ATM)
- Download from a Point of Sale terminal in a shop or service center

In the case of over-the-air reconfiguration by downloading software over a wireless link, the options of point-to-point and point-to-multiple-point are available. This allows forms of broadcast reconfiguration. The Forum anticipates the obvious form of download patterned after Microsoft's Telephone Applications Programmer Interface (TAPI) or the ITU's H.320 video teleconference recommendations. These are based on a capability exchange and on pre-defined levels of capability. The essential steps are defined in the companion Licentiate thesis.

The complexity of a plug-and-play environment for modular radio software raises the question of whether a download is stable. In particular, radio software often has to support isochronous services like voice. In addition to the determinism of the operating system, properties of the applications code can violate stability. In an unstable condition, software consumes unbounded resources in the equivalent of a while loop that has no termination condition. For example, it is possible for two stable modules to be mutually unstable. Consider the following pseudo-code.

Module 1 Set-test(RF)

(Runtime value of RF=30)

Module 2: Test-Receiver(max-rf)

j=150;

While j < max-rf, tune (j); j++; end-while

Module 1 was designed to operate in HF. It sets its RF tuning word to 30 MHz, its maximum value. It then calls Control-Receiver, which was designed to operate in VHF. Module 1 normally calls a Module 2 that loops from the minimum RF (for the purposes of this example, a global that is defaulted if not set) to a maximum supplied by the Set-test function. The assumption on which Module 2 was designed is that RF will never be less than 150 MHz. The assumption of Module 1 is that it will never be greater than 30. When these two modules come together in a software radio, the Set-test function calls Test-receiver with a value, 30, which is less than the lowest assumed value, 150. The while loop that runs the tuning test keeps incrementing forever in search of the termination condition. There are several programming discipline errors committed by the programming team. But such errors happen. Attention to computability reveals provable mechanisms for essentially eliminating these kinds of crashes from radio code (even if the programmers do the kinds of things illustrated above).

5.5.2 Computability Aspects

The architecture paper "Software Radio Architecture: A Mathematical Perspective" was published in the IEEE Journal on Selected Areas in Communications, April, 1999. This paper is an appendix of this thesis. This paper addresses the question of computability of cognitive radio. Cognitive radio, by definition, is first of all a radio. As such, it must deliver services with high quality. This means that it must finish its cognition cycle in time to deliver services isochronously. The architecture paper shows that full Turing computability is not required for isochronous tasks. Instead, the bounded recursive functions are sufficient. The bounded recursive functions preclude unconstrained While and Until loops, substituting instead resource-constrained versions of those functions. In order to do this, one must compute in advance an estimate of the computational resource required for every While and Until loop. The primary resource to be estimated is time. To allow cognitive radio to reason about such resource constraints, RKRL should specific mechanisms for characterizing time allocated to programs, modules, and hardware execution (e.g. despreading). This process precludes crashes due to the use of excessive resources as in the example above. The details are deferred to the appendix.

5.6 Summary of Use Case Analyses

These use cases prioritize models RKRL should express, as summarized in Table 5-10.

Table 5-10 RKRL Model Representation Priorities

Contribution	Mechanism	Models RKRL Should Express
Spectrum Aggregation	Spectrum Rental Etiquette	Spectrum Structure, Propagation
Select Alternate Air Interfaces	Model Bands, Modes	Data rate, code rate, QoS Metrics
Shape Data Demand	Buffer/Burst Email & Files	Space and Time, Delay(message)
Reduce Demand Uncertainty	Demand Source Profiling	Plans, Demand vs. time
Negotiate with Network(s)	KQML using RKRL	KQML, user, network, location
Convenience of Global Travel	Learn protocols for locations	Places, access protocols
Information Queries	Wireless Internet Access	Information needs (e.g. keywords, user expressions of interest), Servers (e.g. Yellow pages), Related points of entry (e.g. information kiosk, local broadcast)
Node Selection Decisions	Match to User Training	Source, QoS, Reliability, Assuredness, Cost Parameters
Context Sensitivity	Recognize Communications Context	Network Traffic, Time/ Space Patterns, Weather
Context Interpretation	Adjust Plans to User Contexts	Position, Travel, User Interests from Speech & Text, A-priori needs, connectedness needs, security needs, intended recipient(s)
Facilitate Certification	Establish Stability under Download by Induction	Computational Resources (Needs, Capacities)

6 The Radio Knowledge Representation Language

This chapter presents RKRL. First, the approach to RKRL is developed using the mathematical framework of point-set topology. Next, an overview of the resulting language is provided. Formal syntax and axiomatic structure is described. Finally, each of the micro-worlds of RKRL 0.3 is summarized.

RKRL has been designed to meet the needs identified by the use cases analyzed above. My licentiate thesis defined the first version of RKRL. The doctoral research refined and extended RKRL through further development of the mathematical foundations and through the implementation of the CR1 rapid prototype. This chapter summarizes the resulting Version 0.3 of RKRL. RKRL is unique in that it employs *heterogeneous model representations in an axiomatic framework based on topological spaces.* The models accommodated by this mathematically precise framework range from natural language (in some sub-spaces) to semi-formal models (e.g. if-then rules) to formal axiomatic models (in other sub-spaces). RKRL's semantics are defined in its computational models. Its syntax, semantics, and initial knowledge are described in this chapter. To enhance access by researchers, the 4,100 frames of RKRL Version 0.3 are available in the Extensible Markup Language (XML) [174].

6.1 Topological Analysis of Radio Knowledge Representation

In order to meet the use-case requirements of the preceding chapter, RKRL has to accommodate heterogeneous semantics. This section shows that topological analysis structures context with the mathematical precision necessary for the control of component-based software radios. This section summarizes how that topological structure is represented in RKRL.

6.1.1 Heterogeneous Semantics

The notion of heterogeneous semantics in cognitive radio is illustrated in Figure 6-1. Consider the semantics associated with symbolic structures in radio engineering implicit in the use cases. The example of the figure lists semantics ascribed to the word-level symbol "signal." Each of the roles cognitive radio could play invokes different semantics for this symbol,

depending on the use context.

Models RKRL Should Express	Symbol => Semantics	Use Context
Spectrum Structure, Propagation	Signal => RF propagation	Propagation Model
Data rate, code rate, QoS Metrics	Signal => Flag set in data bits	Modem Module
Space and Time, Delay(message)	Signal => Message event	Queuing Module
Plans, Demand vs. time	Signal => Part of a plan	Planner
KQML, user, network, location	Signal => Argument in KQML	KQML package
Places, access protocols	Signal => Highway traffic signal	Route following module
Information needs (e.g. keywords, user expressions of interest), Servers (e.g. Yellow pages), Related points of entry (e.g. information kiosk, local broadcast)	Signal => Alert at kiosk	Web browser
Source, QoS, Reliability, Assuredness, Cost Parameters	Signal => Failure indication	Fault recovery module
Network Traffic, Time/ Space Patterns, Weather	Signal => End of superframe	GSM protocol stack
Position, Travel, User Interests from Speech & Text, A-priori needs, connectedness needs, security needs, intended recipient(s)	Signal => INFOSEC alarm	Security module
Computational Resources (Needs, Capacities)	Signal => Semaphore	Diagnostics module

Figure 6-1 Context-Sensitive Implications of the word "signal"

Two aspects of this well-known problem of word-sense ambiguity bear on the formulation of RKRL. First, the user of a PDA may shift among alternative semantics with few syntactic or sentence-level cues. This establishes that RKRL cannot have context-free semantics for at least some of its symbolic structures. The development of natural language processing systems with context-sensitive semantics is in its infancy. Therefore, RKRL contributes to progress in this area by defining context with mathematical precision. RKRL may therefore facilitate the integration of context-sensitivity into wireless applications of natural language processing.

More important for software radio, however, use-context implies the control of software components as listed in the figure. Different software modules may employ the same symbol, "signal" in their control interfaces with different semantics. On small PDR projects with under 100k LOC and a few dozen software components, word-choice in a data dictionary may be tailored to preclude confusion in the project's data dictionary. With the proliferation of software components at all layers of the protocol stack, the likelihood of semantic conflict in data dictionaries increases at least quadratically. This means, as a minimum, that a component-oriented reuse environment faces substantial rework of data dictionaries and risks latent

misunderstandings about module-to-module interfaces that can cause catastrophic system errors.

Explicit representation of word-use context is often used as an ad-hoc mechanism for resolving such ambiguities. In Java and most other programming languages, for example, name spaces may be declared so that each package has a local name space. Similar approaches are taken in data interfaces among SDR components. Suppose the convention is to express a name as **<Name-space>.<token>**. Then, Modem.dataRate may be in units of bps, while UserInterface.dataRate is in units of kbps. More subtly, the user interface may specify the nominal value of the desired data rate, while the Modem.dataRate specifies the estimated value averaged over the past 64 channel symbols. Such semantic differences can be of crucial importance in integrating SDR modules during development and/or in controlling them during operations. RKRL makes these differences explicit so that semantic ambiguities are uniformly identified or resolved algorithmically. This requires a formal treatment of context.

6.1.2 Formalizing Context: The World, World Models, Predicates, and Maps

Researchers in natural language processing, expert systems, and machine learning have recognized the potential advantages of context-sensitivity, but the formulation of context has been illusive. The notion of context has intuitive appeal, but it has no widely agreed to mathematical definition. Creating a rapid-prototype cognitive radio requires a definition that is precise enough to guide the implementation of a specific software system. But creating an architecture for the future evolution of cognitive radio requires multipartite acceptance of a definition that has mathematical precision. This additional precision is needed to assure that teams developing components at different times and/or in different locations use exactly the same semantics for context. They must also know exactly how much precision is associated with each aspect of inexact contexts. This treatment therefore defines context mathematically.

The mathematical analysis of context begins top-down, as illustrated in Figure 6-2. Ordinary experience using a PDA at work, at home, and travelling, as delineated in the use cases, the entails exposure of the PDA to many stimuli. Denote the set of stimuli encountered in the life of a PDA and associated actions taken by the PDA as the set **X**. This is a very large set. Stimuli consist of all of the digitized IF signals received, all of the digitized speech from the PDA's microphone channel, all of the email messages, etc. For example, a PDA that receives

100 messages per day and that supports 30 ten-minute phone calls per day assimilates approximately 90 million word-level tokens per year. If the responses (e.g. transmitting voice, email, etc.) have similar complexity, **X** contains 180 million tokens. This set is large, but not necessarily intractable. A closer look at Figure 6-2 reveals that these stimuli have some structure. Some stimuli refer to home and other places. Some are e-mail from a GSM network. Others are latitude-longitude-altitude vectors from a GPS sensor. These ad-hoc associations establish context but lack sufficient mathematical structure for an axiomatic treatment.

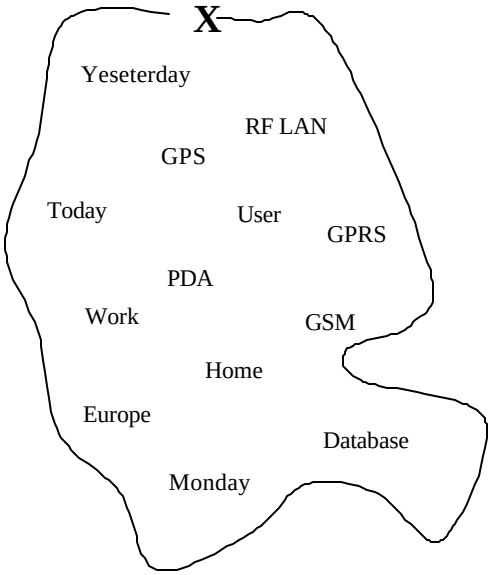


Figure 6-2 Informal Context Associations Lack Mathematical Structure

Figure 6-3 begins to illustrate the mathematical structure of cognitive radio. The PDA must contain a computational model of the world, which should faithfully represent those aspects of the world with which the PDA must exhibit proficiency. This internal world model should map to the external world in a mathematically precise way. Since the internal world model is constrained to be resident in computer memory, it consists of a finite set of symbols, **S**, with a tightly bounded maximum size, **|S|**. In theory of computing, if there exists **|S| < M**, then there exist algorithms that decide properties of **S**. If one cannot specify **M** in advance, then some of the operations needed for cognitive radio are provably undecideable [22]. Given a PDA with 100 MB for control memory, **M** = 100 MB. **S** has a finite number of states, less than $\mathbf{2^M}$. **S** therefore has the mathematical structure of a discrete set of points with bounded maximum size. Although algorithms that compute properties of **S** may require exponential time ("intractable"),

since the dimensions are specified in advance, one can compute whether the property can be determined within an acceptable time budget. For example, if a search algorithm, **f**, requires at most **Ns** steps per item examined in memory, then to search **S** for **s** requires not more than $\mathbf{Nt = Ns*2^M}$ steps. If each step takes at most **t** seconds, and if the answer must be known in **T** seconds, then if **t(f) = t*Nt < T**, one can decide whether **S** contains **s**. If so, then the PDA may compute it and use that outcome. If not, then the PDA knows that it cannot tell. It may so advise its user, the network, etc., so that default or error-correcting action may be taken (e.g. allow more search time, assume the worst, etc.). In any case, one of three outcomes is certain in **T** seconds or less. Either **S** e **s** or **!(S** e **s),** or **t(f)>T**. In other words the PDA decides "Yes", "No", or "Maybe (Can't tell unless I think about it some more)".

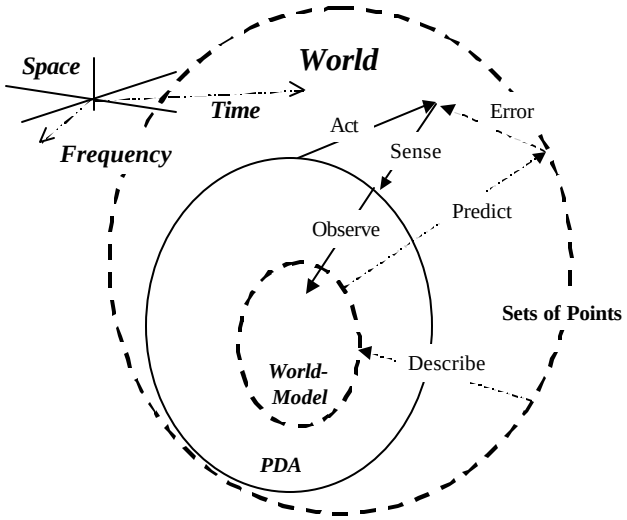


Figure 6-3 Mappings Required for Cognitive Radio

Attention therefore turns to the precision with which **S**, the internal model, can represent the external world, ***W*** [7]. In conventional model-based semantics, predicate valuation functions, **V**, map ***w*** *e* ***W*** onto **s** e **S, V:** ***w*** ® **s**. The structuring of the predicates in contemporary practice is left to the domain expert, knowledge engineer, etc. In natural language processing, properties of ***W*** may be represented as syntactic constraints on **S**. In addition, semantic grammars augment syntactic categories with domain categories, further constraining expressions [78]. This

7 By convention, items in the external world are in ***italics***, while internal symbols are in **bold**.

approach attributes sentential structure to stimuli, which is appropriate for well-structured domains like machine translation. Cognitive PDAs, however, have to perform in the relatively unstructured domains described in the use cases. In addition, a semantic grammar developed for one of cognitive radio's use-context domains may conflict with grammars in other domains. As further illustrated in Figure 6-3, cognitive radio employs multiple mappings to describe the world and to predict the consequences of its actions. Therefore, RKRL needs a top-down structuring of the multiple sub-domains of the use-cases. This top-down structure needs the formal mathematical precision to assure that stimuli, responses, internal inferences, and external effects achieve the performance required of the PDA. These aspects are maps on **S** and ***W***.

6.1.3 Point Sets and Maps in S and W

Since **S** is a point-set, the principles of point-set topology may be applied to study its structure. First, predicates and entity-attribute-value models of object-oriented knowledge engineering are instances of a more fundamental mathematical structure: the point set. Properties of points establish geometric relationships in appropriately defined spaces.

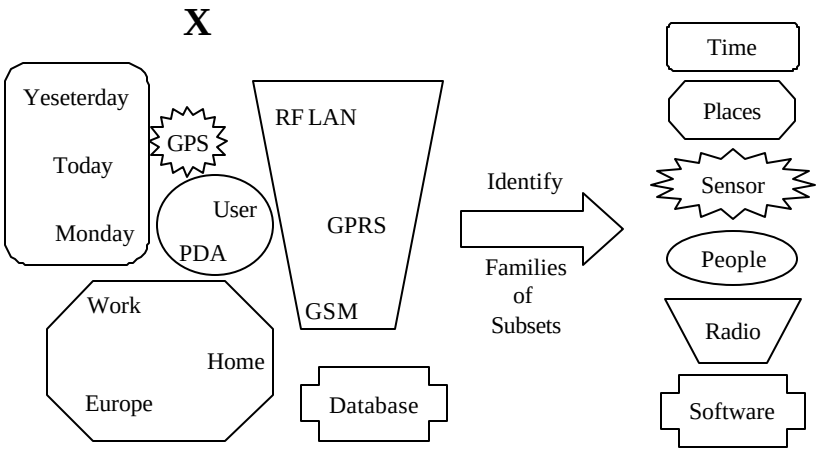


Figure 6-4 Analysis of Point-Sets Yields Families of Subsets

Informally, Figure 6-4 shows how one may identify families of subsets of entities in the world (***w*** e ***W***) by analyzing point sets. Those entities that share points and/or maps among point sets in ***W*** and **S** comprise related subsets. For example, the internal symbols "work", "home", and "Europe" may all refer to points on the physical globe (e.g. **home** e **S** $\ll$ ***home*** e ***Globe*** $\grave{I}$ ***W***). When used in the appropriate word sense, **home** corresponds to a specific address ***home***, while **Europe** corresponds to a region within defined borders of countries. One can therefore

model home, Europe, etc as places on the globe, subsets of **Globe** (**home** Ì **Globe)**. This requires consistent internal and external representations for elements of **S** and ***W***, e.g. $\mathbf{s_{LL}}^{\mathbf{i}}$ = **(Latitude$_i$, Longitude$_i$)** Í **Globe, G:$\mathbf{s_{LL}}$** ® s_{LL} and **H:** $s_{LL} \rightarrow \mathbf{s_{LL}}$. In this case, G and H are identity maps. Thus, the informal notion "home" is formalized in RKRL as a family of subsets of points on the surface of the physical globe (***Globe e W***), the internal representation of those points (**Globe** e **S**), and the related maps.

Maps among temporal point sets must also be treated axiomatically. Consider, for example, the symbols **yesterday, today**, and **Monday** e **S**. **Yesterday** may be formulated as a map from a point set, the real line that represents time, ***T e W***, onto itself, decrementing the day of the week by one. Just like the identity relations G and H for space, internal time **T**e **S** and external time ***T*e *W*** must be related by the identity map **Monday**, similarly, can be viewed as a modulo-7 divisor of ***T***. With this approach, **today**:$\phi \rightarrow$T yields a 24-hour subset of **T.** This subset of **S** increments by 24 hours whenever calendars in ***W*** increment by one day. Referring again to Figure 6-3, the function **today** e **S** predicts the calendar value that one would observe in the world. These symbols are the corresponding subsets of **T**. They form the specific subsets in ***W*** and **S**, named "temporal" in RKRL. In addition, the points of **T** form a compact subspace[8] of ***W***. The symbols for subsets of **T** are more closely associated with each other than points that are not in a common subspace of ***W*** or **S** (such as "Home" or "Money"). Intersections of such point sets, particularly in orthogonal subspaces, create contexts, such as "Home" $\cap$ "Monday".

Other important sets require more complex maps for G and H. Consider a radio's signal in space, ***s(t) e W*** as illustrated in Figure 6-5**.** When the PDA receives the signal, it is converted to internal digital form by an ADC. Thus, the signal is mapped first from the transmitted signal to a received signal, *Receive(s)*, e.g. by the addition of noise and multipath. Next it is digitized by an *ADC* to become the discrete signal **x(i)** e **S**. Thus, the map H from W to S is ADC(Receive(***s(t)***)). Although H corresponds in part to a physically realizable device, the ADC, H ***is not a function*** that transforms signals. H is a map among subsets of **S** and ***W.*** In words, H states: "The analog signal in the time interval surrounding the continuous point ***s(t) e W*** corresponds to the discrete point **x(i)** e **S** at the sampling instant for N contiguous points."

8 A compact subspace is covered by a finite number of subsets; in this case, **T** is covered by **R**.

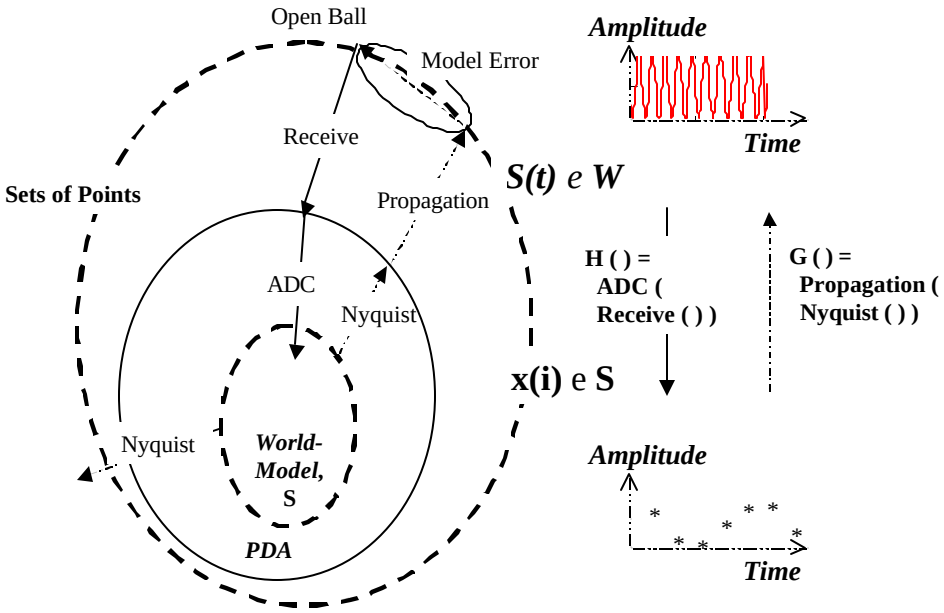


Figure 6-5 Maps G and H for a Radio Signal-In-Space and its Digital Representation

Similarly, the corresponding map G from **S** to ***W*** is not a digital to analog converter. The map G answers the question "How do subsets of **x(i)** map to subsets of ***s(t)***?" This is answered in part by the Nyquist criterion. If a signal ***s(t)*** is sampled at least as fast as specified by the Nyquist criterion, then the analog signal ***s(t)*** may be reconstructed uniquely from the digital samples, **x(i)**, e.g. using Fourier series. In addition, a received signal corresponds to some transmitted signal modified by propagation, so G(**x(i)**) = Propagation(Nyquist(**x(i)**)). G also is not a function. One need not be able to perform G on actual signals. G is the set-theoretic map that specifies which members of **S** correspond to which members of **W.** Thus, G and H are ontological statements about the structure of the point sets ***s(t)*** and **x(i)**.

The general strategy employed in defining RKRL, then, is to identify the point sets in ***W*** and their associated mappings, and to represent them in **S** as corresponding point sets. Both of these ideas may be made mathematically precise as follows.

6.1.4 Inducing Topological Spaces on W and S

The topological structure of a point-set is defined by the family of subsets it induces. In particular, let X be a set and O_X a family of subsets of X. If O_X contains X, the empty set, countable unions and finite intersections, then (X, O_X) is a topological space. Specifically, if 1-4 below are true, then (X, O_X) is a topological space [175]:

1. O_X contains X
2. O_X contains ϕ, the empty set,
3. If B is a countable enumeration and $O_a \in O_X$, then $\bigcup_{aeB} O_a$ is also in O_X,
4. If A is a finite index set and $O_i \in O_X$, then, $\bigcap_{ieA} O_i$ is also in O_X.

The product topology is induced by the set of all subsets of X, which is clearly a topological space. As illustrated in Figure 6-6, it is essential to know what point-sets exist, which ones are relevant to the PDA's communications tasks, and which of these have been characterized. A point-set has been characterized if the knowledge needed by the PDA is represented in RKRL. This knowledge is available to the PDA if it has been mapped from RKRL into the PDA's limited internal data store, **S**.

Topological Space	What Subsets Are Characterized?
O_x Contains X, ϕ	User∩RFLAN: No PDA∩RFLAN: Yes Europe∩Home: No
Countable Unions and Finite Intersections	Europe∩Work: Sometimes?

Figure 6-6 Topological Spaces May Be Induced on W and S

Some subsets, like the intersection of User and RF LAN, may be empty for some purposes (e.g. for communications) and non-empty for others (e.g. measuring health effects of RF radiation). Therefore, RKRL is based on the induction of product topologies. In particular, if $w_1, w_2 \in W$ correspond to $s_1, s_2 \in S$, then RKRL's representation of $\{s_1, s_2\}$ is complete if it enumerates and characterizes $s_1 \cup s_2$ and $s_1 \cap s_2$. Naming a subset enumerates it in the same way that declaring a variable reserves space for it. Characterizing a subset requires the elaboration of the properties of that subset. This is equivalent to providing a program that supplies an appropriate value for the named variable.

Since the size of the product topology is $2^{|S|}$, the explicit enumeration of all subsets of known entities would reduce the content of the knowledge base exponentially for fixed $|S| < M$.

Since many subsets are sparse, in RKRL a set is completely defined when all the non-empty subsets of the discrete topology have been identified. If for $\{s_1, s_2\}$, $s_1 \cap s_2$ is not explicitly in O_X, then this intersection does not exist.

The related knowledge is not complete until the identified subsets have been fully characterized. Thus, when a new entity is introduced to RKRL, its non-empty subsets may intersect with other entities. RKRL axiomatically assumes that any such induced unions and intersections may exist and may be contain relevant knowledge. The incremental development of RKRL consists of exploring and characterizing subsets induced by the introduction of new entities into **S**.

6.1.5 Knowledge as Topology-Preserving Maps

Given the induction of product topologies on ***W*** and **S**, one may characterize the fidelity with which **S** represents ***W***. As illustrated in Figure 6-7, **S** and ***W*** are mapped to each other in general terms by several topological maps.

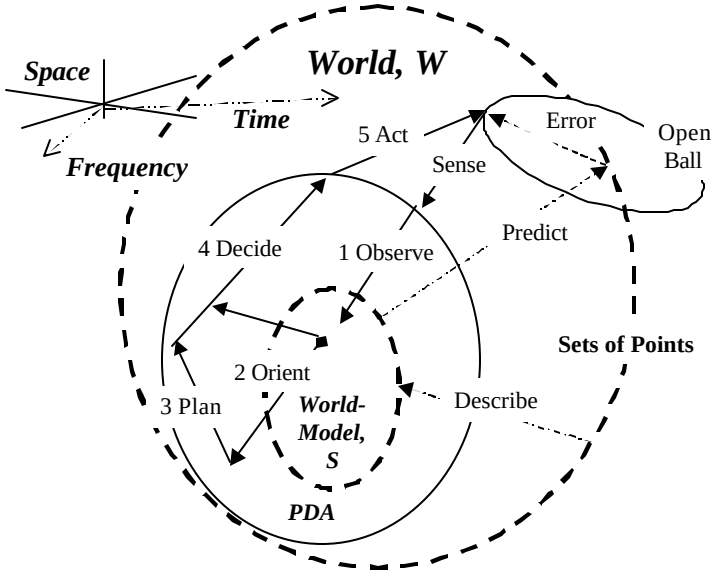


Figure 6-7 Inferences Map S to S, Actions Map S to W, and Sensing Maps W to S

Some of these maps describe the world to the cognitive PDA. The PDA uses others to predict the effects of its actions on the world. In cognitive radio, these maps are realized initially through a core set of knowledge expressed in RKRL and integrated into the PDA. Subsequently, they may be refined and re-defined by the interaction of the PDA with the world. Thus, the five

steps of the cognition cycle, plus sensing, constitute the specific set-theoretic maps by which the world is characterized. Inferences in the cognition cycle map **S** onto **S**. Actions map **S** onto ***W***. The concatenation of sensing and observing maps ***W*** onto **S**. The ability of such maps to preserve the topology of the corresponding point sets is paramount.

A map, f, that is **1:1** from one topological space, X, **onto** another, Y, is *continuous* if the inverse images of open sets in Y are open sets in X. This generalizes the notion of continuity in metric spaces (e.g. the epsilon-delta limit at infinity). For RKRL, this means that a family of subsets of **S** must have a corresponding family of subsets in ***W*** and conversely. For example, time in ***W*** is continuous, but **T** in **S** is a discrete approximation, e.g. to the millisecond. Thus, G and H may not be identity functions. Instead, there may be a map, "*timing*," between **S** and ***W***.

Timing: S® *W*

For each millisecond in ***W***, there corresponds a unique value of the millisecond-counter in **T**. Furthermore, by the product topology, to each union of milliseconds in ***W*** (e.g. the next hour), there correspond a subset of values in **T**. Finally, any subset of values in **T** corresponds to some time interval in ***W***. These values may be reached from ***W*** using $timing^{-1}$. If the internal should clock drift or is not set properly, then the error may be expressed as an open ball, a set of points in ***W***. If the clock is known to be set to within one second of GMT, then "Error < 1000 ms" is that open ball. Since such open balls are equivalent to the more familiar notion of distance, one may precisely characterize the fidelity with which **S** represents ***W*** using open balls.

If a map **f: X® Y** is continuous and has an inverse and if for each open set in **X** the inverse image of open sets is also open, then f is a homeomorphism, a topology-preserving map. Since RKRL represents product topologies, each relationship between **S** and ***W*** can be tested for homeomorphism. If a map is a homeomorphism, then the proven properties of homeomorphism apply. Importantly, the composition of homeomorphisms is also a homeomorphism. Therefore, if all of the maps of Figure 6-7 are homeomorphisms individually, arbitrary compositions are also homeomorphic. For example, if sensing and observing each preserve topological structure individually, then the inferences drawn from the concatenation of sensing and observing correspond to situations in the world *exactly as described* in **S**. Furthermore, if all the maps preserve topology, then situations that develop in ***W*** are reflected in **S**, and inferences drawn in **S**

for actions will yield corresponding effects in ***W***. If the observations conflict with the predictions, then the space **S** and/or the maps need to be refined (e.g. via machine learning). Testing the a-priori correspondences is a matter of checking the RKRL to see that subsets of ***W*** and **S** are defined, and that a domain and range for each map associated with **S** is also defined and has an inverse. If the point set is discrete and all points are mapped in both directions, then the power sets of the points induce the required homeomorphism. Updating dynamic correspondences is the task of the machine learning components of the cognitive radio.

The principles enumerated above establish the topological foundations for RKRL. The pragmatics of realizing this mathematical structure in an intuitively appealing language takes the rest of this chapter.

6.2 RKRL Language Overview

Informally, RKRL consists of the following.

1. The set-theoretic model, introduced above, that encompasses the topological foundations and that admits degrees of axiomatic and non-axiomatic semantics.
2. A syntax defining sets of frames, the statements of the language,
3. Axiomatic models of time, space, entities and communications among entities. Entities occupy subsets of (Time x Space) such as people, places, and things.
4. An initial set of knowledge including initial representation sets, definitions, conceptual models, and radio domain models.
5. Mechanisms for axiomatically modifying and extending RKRL

6.2.1 Overview of RKRL Syntax

Informally, the syntax of RKRL specifies a structured set of statements called frames. Frames are organized into a core set, $RKRL_0$, and an arbitrary number of extensions, $RKRL_j$. $RKRL_0$ consists of exactly three micro-worlds. Its RKRL micro-world identifies the top-level components of RKRL, including the Meta micro-world and the Universe micro-world ε $RKRL_0$. Each additional micro-world, $RKRL_j$, is linked to one of these three micro-worlds through the set-theoretic model ε, *"contains"* as illustrated in Figure 6-8. RKRL 0.3 includes $RKRL_0$ and

about 42 additional micro-worlds.

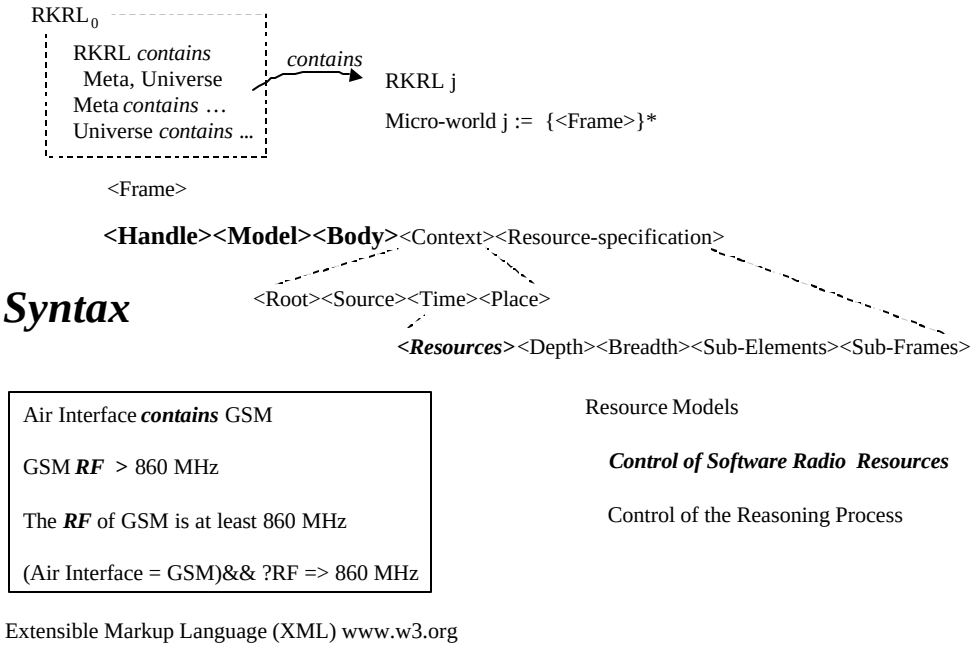


Figure 6-8 Overview of RKRL Syntax

Each frame is a five-tuple that consists of a handle, a model, a body, a context, and a resource specification. Each element of the tuple ("slot") is a String, an array of concatenated characters. The handle names the subset of W or S characterized in the frame. This can also be used to provide access to the frame. The handle is more than the name of the frame, in part, because the same handle may apply to multiple frames. The handles of a micro-world constitute the named point sets and subsets of that micro-world. All handles first appear as the body of a *contains*-frame. Thus, subsets are grown like branches of a tree from the more general to the more specific. For example, the following four (partial) frames name a spatial point set, the Centrum building located in a suburb of Stockholm.

Universe *contains* global plane

Global plane *contains* Europe

Europe *contains* Stockholm

Stockholm *contains* Centrum

The model specifies the relationship between the handle and the body. The body specifies

the contents and/or parameter(s) of the model. Together handle, model, and body specify the knowledge of the domain frame. These three items of information may behave like predicates, neural networks, if-then rules, or cases, depending on how the frame is interpreted. This aspect of RKRL may be called dynamic semantics.

The frame's context specifies the place where the frame fits in RKRL's universal ontology. All context roots terminate with the string "RKRL." This specifies that this frame belongs to RKRL. The context root is a path from RKRL to the frame through set-membership (via *contains*) in the micro-worlds hierarchy of $RKRL_0$ through $RKRL_j$. This path specifies the version of RKRL, and the micro-worlds in order of ascending generality in which the frame may be accessed. Although there is only one context root per frame, a frame may be applicable in more than one context. Context also includes the source of the material in the frame (e.g. the author), and the time and place at which the content of the frame was generated. This additional meta-level frame context supports machine reasoning about the validity of the frame. If the frame is out of date or was provided by a deprecated source, the cognitive agent can take appropriate action to update the frame.

The frame's resources specification includes software-radio domain computational resources (e.g. processing time, computational capacity, memory, and/or interconnect bandwidth). This meta-level knowledge supports machine reasoning about the computational resources needed to accomplish a wireless task. For example, the resources associated with an equalizer could specify the time within which the equalizer should converge. The cognitive radio can apply computational constraints by setting a watchdog timer before invoking an equalizer algorithm. Failure to converge would be detected by the timer so that the cognitive radio could take an appropriate action, such as switching RF bands. Resources need not be computational. Frames concerned with antennas, for example, might specify the maximum RF power that can be supplied to the antenna. The resources specification can be used to assign resources to air interfaces, user interactions, etc. The degree to which non-computational resource constraints (e.g. RF power) can be controlled depends on the degree of instrumentation of the radio hardware platform. The computational resource specification is necessitated by the mathematical analysis of Appendix B. Without such constraints, cognitive radio could not reliably reason about downloads, for example. The additional resource constraint enforcement is

needed to insure that the radio software operates within hardware constraints.

The resources-specification of the frame also identifies reasoning resources in terms of the maximum breadth, depth, elements, and frames contained within the subordinate sets linked to the frame. This enables machine reasoning about resources needed for performing inferences using the models of the micro-world. Reasoning tasks include parsing stimuli and context-sensitive planning during wake epochs and learning during sleep epochs. A cognitive radio may employ this meta-data using the resource-bounding algorithms exhibited in Appendix B. This can provide an upper bound on the duration of sleep epochs (e.g. so that it will not be in the middle of a computationally-intensive model update cycle - asleep - when the user is next expected to use the PDA).

6.2.2 Heterogeneous Knowledge Representation

In most machine reasoning systems, knowledge is represented homogeneously. Rule-bases all use rules, predicate calculus systems use Horn clauses, typically interpreted by a Prolog engine. Case-based reasoning systems tend to use a data base structure, as do many decision-tree systems. Object-oriented technology makes it easy to employ multiple representations for the same information by attaching alternate representations to the object's slots. But Smalltalk objects, Java objects, and C++ objects are all declared differently. RKRL is a computer-language independent representation of the set-theoretic and model structure of such objects, mappable to any of them. It supports heterogeneous knowledge representation. In particular, as illustrated in Figure 6-9, frames may incorporate one of the following model classes:

Model Class	Interpretation Mechanism
1. Ontological (e.g. Contains, Set)	Set theoretic
2. Natural Language (e.g. Definition …)	Human interpretation of language
3. Axiomatic (e.g. Time, Now, Place, Location …)	Preservation of axioms
4. Stimulus-Response (the default interpretation)	srModel interpretation
5. Reserved (e.g. Excel, Rule, Predicate, Plan …)	Reserved for interpreters

RKRL does not specify that one form is preferred over another. However, it does specify

that if a string is used as a model, then that model has to be identified in the Models meta-world. Space, time, RF (e.g. radio signals and propagation), and entities are axiomatized.

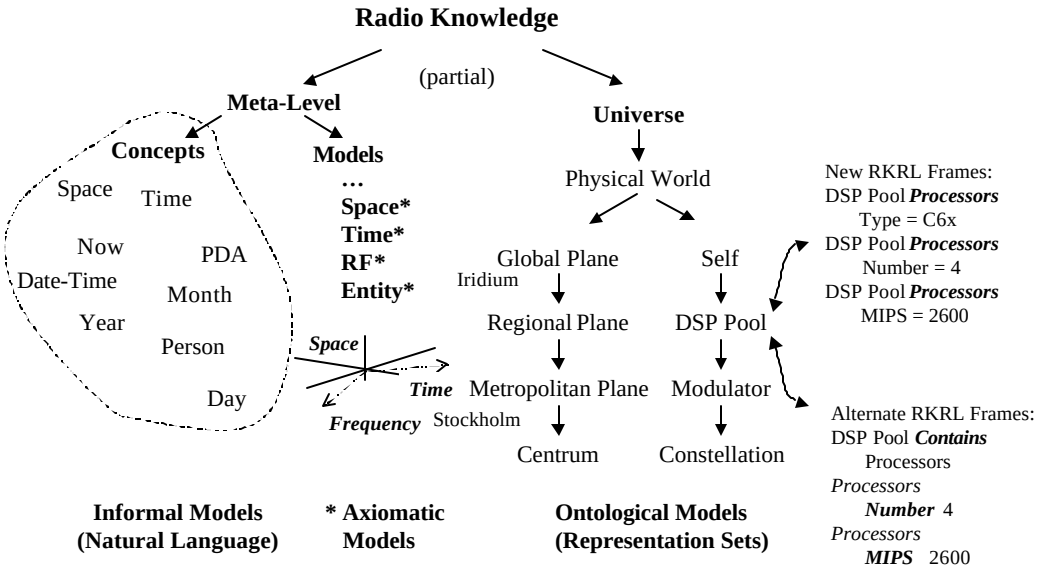


Figure 6-9 Heterogeneous Knowledge Representation in RKRL

In addition, the semantics of that model has to be specified somewhere in the inference hierarchy, e.g. by specifying the API. Each model slot in an RKRL frame implies the related interpretation mechanism. For example, the *contains* model entails set-theoretic expansion of the parent frame to contain child frames that inherit the parent's context. Natural language models rely on human interpretation of language. These may be as formal as the KQML micro-world or as informal as the User micro-world, as the use context indicates. Models of time and space have to be preserved by the user as axiomatic. In particular, the interpretation environment must obey the associated axioms. If the model is "reserved," a specific model interpreter is needed, such as a Prolog engine. The srModels are bound to stimuli to yield responses as described subsequently. Because of RKRL's syntax, any of these models may be interpreted as srModels. That is, any frame may be interpreted according to the following pseudo-Java interpreter:

srInterpret (stimulus, context, resources, resources-available; Micro-world){
    If(resources-available > resources) {
        Return(getBody$_{\text{Micro-world}}$ (findHandle(stimulus, context)));
    }else{Return "I need more resources to investigate"+stimulus;}}

This interpretation treats each frame as an srModel. This has some advantages in

representation flexibility. For example, Figure 6-9 includes abbreviated frames elaborating on the DSP pool. In the first approach, the DSP Pool is described in terms of a model called ***processors***. With this approach, all of the new frames are part of the ***processors*** model. To find out the processor type, one must sort through the list of responses returned from presenting "DSP Pool" as a query to the ***processors*** model. This can be advantageous if one wants all the information about the DSP Pool at once.

Alternatively, one could represent the same knowledge with the statement that the DSP Pool ***contains*** processors. This establishes processors as a named subset. Other frames with processors as the handle may elaborate. The context for any subsequent processors frame is DSP Pool/<prior context>. Therefore, it is clear from the explicit context that the content of the frame refers to the DSP Pool. A frame may state that "Processors ***number*** 4." In this case, the model semantics is "the ***number*** of processors is 4". The frame behaves like an entity-attribute-value object. The processors (of the DSP Pool) is the entity, the attribute is ***number***, and the value is 4. Thus, <model> specifications can behave like attributes, <body>s like values, and <handles> like entities. The more general expression of the semantics is "the attribute of the entity is value" or "the <model> of the <handle><context> is <body>." A practical RKRL interpreter (e.g. CR1) could consolidate these frames into srModel: DSP Pool, with frames denoting the following values of attributes of the processors entity: (processors, number)->4, (processors, type) -> C6x, etc. The source frames would be:

Efficient srModel:

Handle	**Model**	**Body**
0. Digital Hardware	Contains	DSP Pool
1. DSP Pool	Contains	Processors
2. Processors	Type-Query	(Processors, type)
3. Processors	Number-Query	(Processors, number)
4. Processors	MIPS-Query	(Processors, MIPS)
5. (Processors, type)	DSP Pool	C6x
6. (Processors, number)	DSP Pool	4
7. (Processors, MIPS)	DSP Pool	2600

In this style of populating sets into RKRL, frames descend the ontological hierarchy until they encounter an atomic unit of knowledge. "There are four processors in the DSP Pool." This

representation permits an efficient srModel to be expressed in RKRL and run using the simple srModel interpreter ***srInterpret***. In addition, RKRL's set-theoretic foundation provides a natural way of deriving effectively-computable models. In frames 0 above, the DSP Pool was named as a subset of Digital Hardware, presumably of a PDA. At this point, the subset has been enumerated but not characterized. In frame 1, the additional subset of DSP pool, processors, is enumerated. Frames 5-7 use DSP Pool as the model within which the properties of the processors are defined. This is natural since processors is a named subset of DSP Pool. The point at which DSP Pool becomes a model is the point at which it is characterized. A knowledge acquisition algorithm can easily determine which named subsets exist in S. It can then check to see which of these is not used as a model in a frame. If there are none, but that named subset becomes relevant to an inference in S, then it can ask the system and/or the user to tell it about that subset, creating a model. The basic techniques would be those of Davis schema-schema [35]. If number of processors, type, and MIPS are the only subsets enumerated for the set Digital Hardware, then the characterization of DSP Pool would be algorithmically complete.

The named subset offers additional interesting ways of partitioning knowledge. For example, knowledge about queries can be kept at the concept level of RKRL, as follows:

Frames for Query Schema as a Concept

Handle	**Model**	**Body**	**Context Root**
1. RKRL	Contains	Concept	
2. Concept	Contains	Query	RKRL
3. Query	Contains	Entity	Concept/RKRL
4. Query	Contains	Attribute	Concept/RKRL
5. Query	Contains	Class	Concept/RKRL
6. Query	Contains	Value	Concept/RKRL
7. Query	Contains	DeclarationFrame	Concept/RKRL
8. Query	Contains	Value Frame	Concept/RKRL
9. Declaration Frame	Contains	<Handle>	Query/Concept…
10. Declaration Frame	Contains	<Model>	Query/Concept…
11. Declaration Frame	Contains	<Body>	Query/Concept…
12. <Handle>	Sequence	Entity	Declaration Frame …
13. <Model>	Sequence	Attribute-Query	Declaration Frame
14. <Body>	Sequence	(Entity, Attribute)	Declaration Frame

These frames constitute a detailed schema for the declaration frame of an arbitrary query. The frames 2-4 of the DSP Pool model conform to this schema and could have been interactively

generated from it. A few additional frames added to the Query Schema define the value frames (frames 5-7) of the DSP Pool model, as follows:

Additional Frames for Query Schema Define Value Frames of a Query

Handle	**Model**	**Body**	**Context Root**
15. Query	Contains	Value Frame	Concept/RKRL
16. Value Frame	Contains	<Handle>	Query/Concept…
17. Value Frame	Contains	<Model>	Query/Concept…
18. Value Frame	Contains	<Body>	Query/Concept…
19. <Handle>	Sequence	(Entity, Attribute)	Value Frame/Query…
20. <Model>	Sequence	Class	Value Frame/Query…
21. <Body>	Sequence	Value	Value Frame/Query…

This completes the definition of meta-level frames needed to precisely model the informal notion of Query as a characterized subset of DSP Pool. Note also that the inclusion of the contex root in the frames precludes namespace conflicts. These frames act like a (serialized) class specification. Since they exist at the Concept level of the RKRL taxonomy, they are available in any context, but they can also be overwritten in lower-level contexts without confusion. For example, there might be a context "SDR Forum/ Query/ Concept" that overrides this default specification. This alternative could be employed by those who wish to conform to the standard offered by the SDR Forum. Or there might be a context " My PDA/ NexTel/ Washington DC/ North America/ Universe/ RKRL" that specifies the way NexTel structures this knowledge in the Washington, DC metropolitan area. Since the frames are available in XML, they could be interpreted in Java, LISP, C++, Smalltalk or any other present or future development environment with object-oriented capabilities. RKRL, thus, is an extensible language for the evolution of software-defined radios.

6.2.3 Overview of the Axioms

The primary set-theoretic axiom of internal representation consistency is that every RKRL frame is a member of the set RKRL or one of its subsets. This is the fundamental axiom of RKRL as a topological space as formulated above. All frames are therefore derived from other frames via the *expand* operator. *Expand* assures that a body of some frame *contains* the new elements (members or named subsets). This axiomatizes the property that all frames are linked to the root frame containing only "RKRL." Multiple frames may have the same handle but

different models, bodies, etc. It is axiomatic that the context root of each frame expresses a path from the current frame to the root, "RKRL." This permits one to express a variety of models (types of information) that refer to the same concept, entity, attribute, object, situation, context, etc. The primary model-reference axiom is that there is a mapping between the sets represented in RKRL and sets occurring in the world. The fundamental axiom of the frame is that the semantics each frame is specified in the frame's <Model>.

Each frame may have natural language content, e.g. using natural language models "Definition" or "Word Role." Other frames may serve only the set-theoretic function of referring to sets of other frames. Alternatively, the frame may be formally defined in terms of axiomatic models. Space, time, and objects that occupy (Time x Space) obey axioms. Space is represented using two overlapping coordinate systems. Location is specified by latitude and longitude, and optionally elevation and velocity (speed in each direction), e.g. from GPS coordinates. If only latitude and longitude are available, reasoning is required to infer altitude as a function of context (e.g. at aircraft altitude if the user's itinerary calls for a flight, or at ground level otherwise). The error associated with such location estimates is also axiomatic, and may be a function of location and speed. For positioning inside buildings, positions are axiomatic. Position sensors read (x, y), (x, y, z) tuples, or the names of landmarks that are advertising their presence (e.g. "The Front Door to the Centrum Building.") Objects that occupy (Time x Space) may be people, places, things, concepts or hybrids. Fore example, a river is a thing, but Germany is a place concept having physical extent specified by its borders, a set of (Latitude, Longitude) points. Space, time, person, place (location and/or position), thing, temporal concept and spatial concept are the logical sorts governed by these axioms. A frame that mentions "thing" in its model slot declares that the handle and body participate in the "thing" model. For example, the frame:

Antenna *thing* Radiates RF energy,

specifies that an antenna (handle) is something (model) that radiates RF energy (body). If an algorithm wants to know if an antenna could be a *thing*, it presents the string "antenna" to the *thing* model in the appropriate context, and inspects the result. If it is null, then the system does not know of antenna as a thing in that context. The user may then search the RKRL hierarchy for other knowledge about antennas. Alternatively, the handles of the RKRL frames may be

searched for “Antenna” to identify the contexts in which it occurs.

The behavior of objects that emit and reflect radio waves is axiomatic. The axiomatic treatment of radio propagation is that the instantaneous field vector at any point may be calculated from Maxwell’s Equations. All objects in a scene, therefore, may interact with radio waves. Waves may be generated, reflected, diffracted, and/or absorbed. Each function that calculates the field vector is therefore, axiomatically, an approximation of Maxwell’s equations. This admits high-fidelity approximations like the Geometric Theory of Diffraction (GTD) [176, 14, 177]. This also allows computationally intensive approaches like ray tracing [178, 176]. On the other hand, much simpler, less computationally-intensive, empirical techniques are also admitted that approximate received signal strength based on a few parameters [179, 180, 181].

The behavior of most modeled objects is expressly not axiomatic. Except for the occupancy of space and time and radio propagation, any object may behave in any observed way. The statements of a specific micro-world may constrain or intensionalize behavior. If so, that is a property of the micro-world admitted by RKRL but not imposed by RKRL. This is an important ontological stance. Intensional models, like expert system rules or functional grammars, impose structure on a domain. RKRL may be used to express that structure if it is present, but RKRL can express knowledge extensionally if structure is not present or is not yet formulated.

Finally, the axiomatic treatment admits extensions that are set-theoretic but that do not conform to the other axioms. In other words, arbitrarily represented knowledge may be expressed by *contains* with appropriate models in an RKRL frame. This is useful in building up a dictionary of stimuli to which a cognitive radio has been exposed but about which it has no further information. The list of such unknown stimuli can be used as a basis for interactive knowledge acquisition with the user, network, or designer.

Micro-worlds are built up using a mix of axiomatic and non-axiomatic sorts. They may be built up as cases through the experience of cognitive PDAs, for example. Other micro-worlds are designed for specific purposes, such as defining a high level or novel data exchange protocol. Low level encapsulated protocols like TCP, GPRS, etc. are managed by cognitive radio and thus are expressed as parameter sets defined over an Applications Programmers Interface (API) in

RKRL. The details of TCP usually would not be expressed in RKRL frames. The modules of a dynamically defined protocol, on the other hand, would be expressed in RKRL and manipulated by cognitive radio, e.g. in tailoring a protocol to an application. Frames may also specify state machines and message sequence diagrams using the phrase-structure grammar developed for CR1. This might not be the most computationally-efficient way to realize a protocol, but it is flexible in order to mediating higher level tasks such as selecting from among protocols, setting up protocol parameters, and the like.

6.2.4 Interpreting Frames

Each frame can be interpreted as an srModel in which the handle is matched (resolved or bound) to a stimulus and the body is the response of the model. If the frame is purely extensional, the body is the response. If the model is partially intensional [28], the binding of a stimulus to the handle returns model (handle, body, context, resources). That is, the handle is bound and passed to the model (optionally with the context and/or resource limits), returning a result that has the form specified in the body. As shown in Figure 6-10, this permits RKRL frames to be interpreted as propositions, genetic structures, expert system rules, predicates, Horn clauses, cases, or a data base.

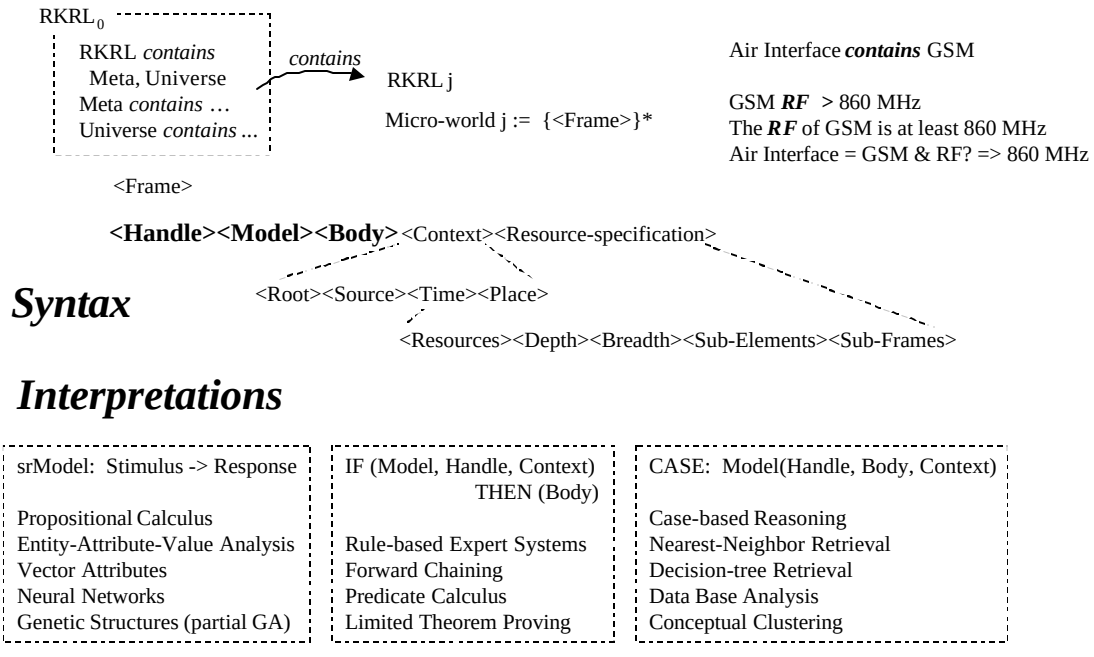


Figure 6-10 RKRL Frame Interpretation Depends on the Model and Context

This approach enables heterogeneous knowledge representations to co-exist in a single XML syntax. Instead of interpretation using a single style implicit in the use-environment, RKRL makes the mode of interpretation explicit as the model of the frame. For example, if the model is "RULE" the handle may be interpreted as the antecedent of an if-then rule with the body as its consequent. A different rule-based inference engine might want the handle to represent a unique name for each rule and the body to represent the antecedent and consequent of the rule. That is not precluded, provided the model is explicit, e.g. the rule model is Nexpert-Rule. Other frames may be interpreted as problem-solution cases in which the context and handle specify the model aspect of a problem, and the body specifies the solution.

6.2.5 Incremental Extensions

Frames may be added by asserting a frame for which the model is *contains* and the body is the new item. That body then may be the handle of a new frame that further defines the new entity or concept. If not, then the entity is one of the leaves of the RKRL taxonomy, functioning as a member of the set indicated in the handle. Consider the following two abbreviated frames:

	<Handle>	<Model>	<Body>	<Context>
(1)	Protocol	contains	UDP	Networks/Concepts/Version 0.3/RKRL
(2)	Protocol	contains	TCP	Networks/Concepts/Version 0.3/RKRL

These two partial frames say that protocols are network concepts and they include (but are not limited to) UDP and TCP. The details of TCP would be provided in amplifying frames. In addition, the UDP-defining frames might also be reachable from the context Telia/Stockholm/Europe/Earth/Universe/Version 0.3/ RKRL if Telia's uses UDP on a network in Stockholm. Frames from that context might differ from frames in the Networks/Concepts context, overriding them as appropriate to the local conditions of the network. Thus, the addition of frames to RKRL can achieve the same effect as overriding a slot or method in an object-oriented representation. The frames that are closest to the invocation context will be used first, overriding similar frames elsewhere in the hierarchy. Frames may be represented as serialized Java objects, and conversely.

RKRL is therefore an extensible language for representing radio knowledge in a way that is

independent of software operating environment, but mappable to any object-oriented, rule-based, or logic-programming target use platform. In addition, its set-theoretic foundations are based on point-set topology, not the valuation functions of the predicate calculus. This brings mathematical precision to the relationship between entities in the real world, W, and the internal world of the software radio, S. The identification of point sets, the evaluation of the homeomorphism properties of behaviors that map among W and S is unique to RKRL. In addition, RKRL thus formulated has the flexibility to address many aspects of the use-cases in ways that are awkward at best using any of the homogeneous approaches. Finally, since RKRL is available in XML frames, it is a starting point for an open-architecture language for the rapid evolution of software radio. Important details are now discussed.

6.3 Syntax

The syntax of RKRL is as follows.

1. RKRL := $RKRL_0 \cup RKRL_j$
2. $RKRL_0$:={<Frame>}$_0$

That is, RKRL is the union of an initial set of frames, $RKRL_0$, and extension(s) $RKRL_j$.

A frame is a tuple:

3. <Frame> := [<Handle> <Model><Body><Context><Resources>]
4. <Handle>, <Model>, <Body>, <Context>, <Resources> ε String.

Each frame is defined in the context of a set S of frames (e.g. $RKRL_0$, $RKRL_1$, etc.) as follows:

5. ∃ S = {<Frame>*}, and for each frame F_i ε S.
6. $<Frame>_{j+1}$ is derived from $<Frame>_j$ ⇔

   $<Handle>_{j+1} = <Body>_j$; and $<Context\text{-}root>_{j+1} = <Handle>_j/<Context\text{-}root>_j$

The predicate Link($<Frame>_{j+1}$, $<Frame>_j$) is true in 6. In addition, Link is transitive and inherited by sets of frames:

7. Link($<Frame>_x$, $<Frame>_y$) and Link($<Frame>_y$, $<Frame>_z$)

$\Rightarrow$ Link(<Frame>$_x$, <Frame>$_z$)

8. Link({<Frame>$_x$}, <Frame>$_y$) $\Leftrightarrow$ Link(<Frame>$_x$, <Frame>$_y$).

The following axioms apply to the structure of all frames:

9. Hj is a <Handle> $\Leftrightarrow$ $\exists$ F_i ε S such that H_j = <Body>$_i$ and <Model>$_i$ = "Contains" and
10. Mj is the Model of F_j $\Leftrightarrow$ $\exists$ F_i ε RKRL such that M_j= <Body>$_i$ and <Model>$_i$ = "Model"
11. <Context> := <Context-root>, <Source>, <Time>, <Place>, where:
12. <Context-root>$_j$:= RKRL | <Handle>$_{j-1}$/<Context-root>$_j$
13. <Source> ε String and <Source> specifies the source of the frame
14. <Time> ε String and <Time> specifies the time of creation of the frame
15. <Place> ε String and <Place> specifies the place of creation of the frame
16. <Resources> := <Model-resources>, <Depth>,<Breadth>, <Sub-elements>,<Sub-frames>
17. <Depth> is the maximum number of elements in any <Context> linked to <Frame>
18. <Breadth> is the maximum number of frames that have a mutual handle
19. <Sub-elements>$_y$ is the number of frames for which <Handle>$_x$=<Body>$_y$
20. <Sub-frames>$_y$ is the number of frames for which Link(<Frame>$_y$, <Frame>$_x$) is true
21. RKRL+ := {<Frame>}* such that $\forall$ $Frame_i$ ε RKRL +, Link($RKRL_0$, $Frame_i$).

The following summarizes the import of these axioms. All elements of frames are representations, strings. The model expresses the relationship between the handle and the body. The context root specifies the most general point of insertion of this statement in the tree of representation sets. Extended context also specifies the time and place of the entry and its source. The resources specify the software radio resources needed to perform the operation represented in the frame, if any. This includes the computational burden associated with the frame. If the frame is purely declarative (contains no computational model), then resources may be null. Micro-worlds are collections of frames. Each micro-world contains at least one frame.

6.4 General Logical Sorts and Axioms

The logical sorts and axioms of RKRL implement the mathematical foundation of RKRL in point-set topology. The identification of point sets and subsets in RKRL is the primary paradigm for knowledge organization in RKRL. The hierarchical inheritance of object-oriented approaches is replaced by inference over the RKRL ontology of sets, subsets, and super-sets. A *class* is a set that is interpreted as a template for another set. Thus, any set may be interpreted as a class. Objects are encapsulated by associating related data structures (slots) and procedures (methods) to a class. An instance then inherits slots and methods from the class. RKRL's representation-sets give names to slots and methods in frames. Slots and methods, then are interpretations of RKRL frames.

6.4.1 Ontological Expressions

RKRL sets and subsets are enumerated via a *contains* frame.

[<Handle> *contains* <Body><Context><Resources>]

If such a statement is present in {<Frames>}, the set of frames in use, then <body> is an enumerated point-set or subset of a point set. The <body> may appear as a <Handle> in other frames that characterize the set or subset. In RKRL 0.3, for parsing pragmatics, the name of a point set or subset must not include white space. Any <body> that contains white space may not be part of a ***contains*** frame. The following sequence of frames establishes Europe and the Atlantic Ocean as named subsets of the physical world.

Handle	**Model**	**Body**	**Context (Place Time Source)**
RKRL	***Contains***	***Universe***	***Version 0.1***
			(Fairfax, VA 9904282137 J. Mitola III)
Universe	***Contains***	***Physical World***	***RKRL/Version 0.1 (Fairfax…***
Physical World	***Contains***	***Global Plane***	***Universe/RKRL/Version 0.1 (Fairfax…***
Global -Plane	***Contains***	***Europe***	***Physical-World/Universe/RKRL/Versi…***
Global-Plane	***Atlantic Ocean***	***Contains***	***Physical-World/Universe/RKRL …***

(Since the contexts are tedious, the obvious higher levels are implied in ellipsis for clarity)

Each of the other major geographical regions of the world is included in the Global-Plane

in RKRL 0.3.

Handle	Global Plane
Model	***Set***
Body	(Africa Alaska Atlantic Australia Canada Caribbean China Europe GEO A GEO B GEO C GEO D HEO Orbit India Information Interconnect Japan Low-Earth-Orbit Mid-East Pacific Russia South America South Asia Temporal Pattern Travel USA)
Context	Physical World Model/Universe/RKRL …
Place:	Rehoboth, DE
Time:	7/25/99 11:22
Source	SetAccumulate()

Figure 6-11 SetAccumulate() Yields the Set Frame

The model *"**set**"* is reserved. In RKRL, set-accumulation lists the enumerated subsets of a named set. Applying set-accumulate to Global Plane yields the associated RKRL ***set*** frame of Figure 6-11. In natural language this frame is read "The Global Plane set is (Africa, …) in the Physical World Model, according to the SetAccumulate() operator of RKRL that was run on 7/25/99 at 11:22 AM in Rehoboth, Delaware." A CR1 planner, for example, can find how the world is divided into regions from this ***set*** frame. Its extended context permits reasoning about whether it is authoritative or might need to be updated.

Definitions are formal RKRL models that establish relationships among natural language expressions. For example:

Handle	**Model**	**Body**	**Context (Place Time Source)**
And	***Definition***	Logical Conjunction	Models/Universe/RKRL Version…

This frame in the Models micro-world defines And to be a logical conjunction. (Statements from RKRL 0.1 show how to compute the conjunction using Excel's AND operation; 0.3 supports Java). Such definitions allow a user to accompany an RKRL computational models with natural language statements, e.g. documentation. In other cases, such as the following, it helps to describe general and specific knowledge about radios:

Handle	RF Channel
Model	Definition
Body	Epoch in radio frequency allocated to a single carrier for a specified time interval in a specified place
Context:	Air Interface/Radio Functions/Universe/RKRL/Version 0.1
Place:	Rehoboth, DE…

Handle **Model**	Carrier Bandwidth Definition
Body	200 kHz
Context:	GSM/Air Interface/Radio Functions/Universe/RKRL/Version 0.1
Place:	Rehoboth, DE …

Through the *definition* model, RKRL has a way of going incrementally beyond ontological awareness. These chunks of knowledge can be played back to users and/or networks in the interactive acquisition of incrementally more formal models. A production-quality RKRL interpreter (e.g. an extension of CR1) would employ definition frames to interactively characterize point sets.

6.4.2 Axiomatic Model of Radio Communications

In Shannon communications, a source presents information to a noisy channel that delivers the information to a receiver. The model of Figure 6-12 expands this model to an axiomatic treatment of users and context. The formal objects consist of physical space and time, originator, recipient, node(s), and channel.

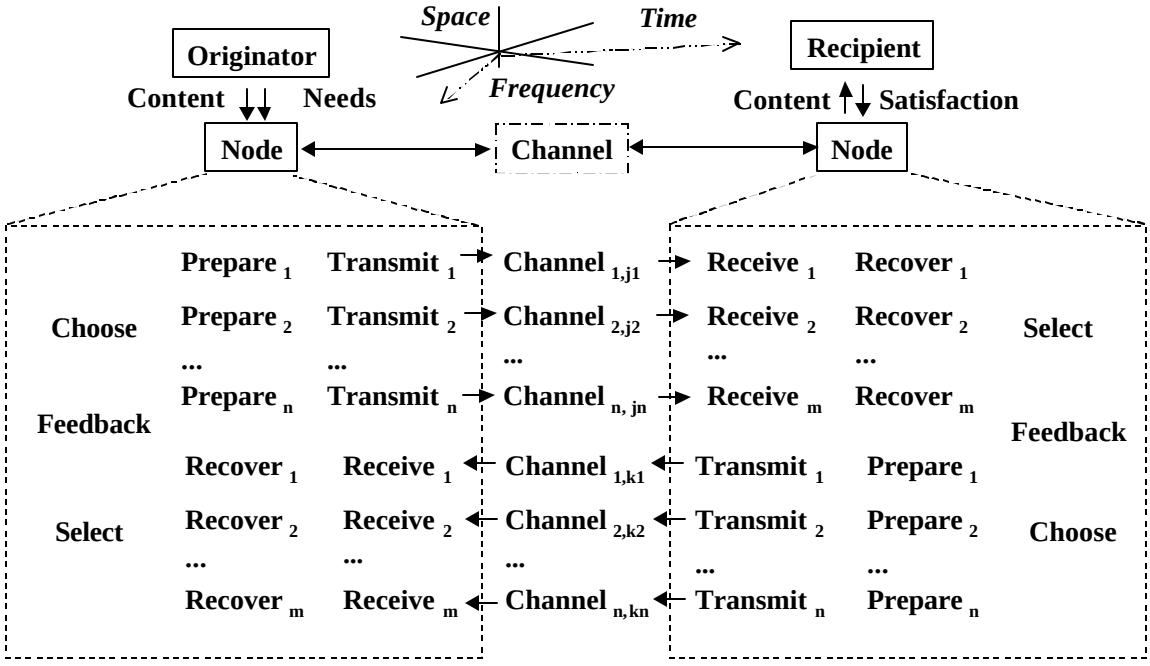


Figure 6-12 Model of Radio Communications

People and radio nodes are distinguished objects. People are the only objects that create

data that contains information, while only radios can transmit and receive signals, which are signals over time, space, and frequency. People may create devices such as radio beacons that autonomously transmit signals, but in this formal treatment, the information content originates from a person (or a group of people such as the Federal Aviation Administration). Each cognitive radio has a global list of things that it knows. " The logical sorts that formalize these statements are provided in Table 6-1.

Table 6-1 Logical Sorts for Distinguished Objects

Logical Sort	**Notation**
The World	**W**
The Model of the World	**S, |S| < M,** a finite upper bound known a-priori
Universe, the set of all things in the universe	**U ⊆ W; S(U) ⊆ S; f_U:U® S(U)**
Person	**P Í W; S(P) Í S; f_P: P ® S(P)** see below
Node	**Node Í W; S(Node) Í S; f_{Node}: Node ® S(Node)**
Concept, an abstraction that is formally defined	C; C ⊆ **U**
The Integers	**Z; Z** ε C
Sequence	$X_Z \subseteq X$ x **Z**
Signal	X(t) ⊆ **C** x T; **C** are the complex numbers
Information	C ⊆ I; I* ε C (I is represented as a concept)
Message	M ⊆ I; M* ε C
Stream	Stream ⊆ M x **Z**; Stream* ε C
Maps	Map ⊆ IxIx…I, a finite Cartesian product of I
Data	D ⊆ O; D* ε C
State	M ⊆ O, memory, aspects of an object that are persistent
Instruction (of an Instruction Set Architecture)	i ε ISA ⊆ I x D
Program	P ISA x **Z**
Channel (in a spatial zone Z)	$C_Z \subseteq$ E x T x RF, C_Z* ε C
Radio Noise	η ⊆ F x RF; η* ε C
Event	V = (t, X ⊆ O x E x T x RF); V ε C; |V∩T|=0
Fluent	F ε V, such that |F∩T|>0
Transmit	Transmit: S → Cz
Receive	Receive: Cz →S
Inference	Inference: S→S
Prepare, Choose, Feedback, Select	Prepare, Choose, Feedback, Select ε Inference
*Formal representation	X* is the formal representation of X

The Universe, **U Í S**, is the set of everything known about the universe. C is the set of

concepts that have been formalized. It is not possible to put all the information in the universe, I, into $C \subseteq S$. But the formalization of all that information, I*, *is* on this "list. None of the items in in C, but their formal representations are in C. This treatment consistently distinguishes between point sets in W and representation sets in S. Thus, a context, X, occurs in the universe, but only X*, the representation, is in C.

6.4.3 Space, Time, and Physical Objects

Space, time, and physical objects are formalized in Table 6-2. Predicates map to TRUE, FALSE, UNKNOWN, and (CHECKING, T). Unknown means that the predicate cannot determine whether it is true or false given the information in S and current resource constraints. Checking means that the information is not available locally, but the predicate knew who to ask (e.g. the network) and has initiated a query with maximum time allowed of T. Such predicates permit processes to proceed in a wait-free way with incomplete information. A Java thread, for example, might suspend itself until T. People, radio nodes, and computational elements have axiomatic properties as shown in the table.

Table 6-2 Logical Sorts for Space, Time and Frequency

Logical Sort	**Notation**
Point	w
Set	W; w is a member of W is noted $w \varepsilon W$
The empty set	ϕ
Space, a set of points in 3-space	E, Real Euclidean space, $\exists e_0 \varepsilon E$
Region or Zone	$R \subseteq E$, or $G, \subseteq E$, the geodesic subset of E
Somewhere, an arbitrary location	E_X; $E_X \varepsilon E$
Physical Objects	$O \subseteq E$; $o \varepsilon O$ is closed and compact
Person	$P \subseteq O$, and $p \varepsilon P$ is closed and compact
Radio Nodes	$RN \subseteq O$, and $r \varepsilon RN$ is closed and compact
Computational Element	$CE \subseteq O$, and $c \varepsilon CE$ is closed and compact
Time, a set of points orthogonal to E	T; $t \varepsilon T$ is an instant
Time Interval (or Period)	$T_{I,j} \subseteq T$
Context, a confluence of space, time and state	$X \subseteq T \times E \times M$; $X^* \varepsilon C$
Radio Frequency	RF, or $f_I \varepsilon RF$; $RF \perp E \times T$; $RF^* \varepsilon X$
Map	$M \subseteq W \times W \times \ldots W$, for any set W

Additional non-logical primitives axiomatize the behavior of space (Table 6-3) and time (Table 6-4). The predicate ***near*** axiomatizes proximity. The Physical Object predicate determines whether x is known in S to be a member of the physical object family of subsets.

Table 6-3 Spatial Primitives

Spatial Primitive	**Characteristics**
Location(x) function	Latitude and Longitude of x, with error σ
Position(x) function	Local coordinates (x, y) or (x, y, z) within Location(x)
Near(e_1, e_2, b), predicate; b ε **R**	Location e_1 is within b of e_2
e ε R, predicate	Location e is in spatial region R
R1 $\subseteq$ R2, predicate	Spatial region R1 is a subset of R2
Physical-Object(x), predicate	x $\subseteq$ E, x is not empty, closed, bounded, and equal to the interior of its closure
Locale(x), predicate	Space sensed by x; Locale(t,p) is the space sensed by Self at time t from position p = (x, y, z)
Intersection(R1, R2), function	Yields the intersection of R1 with R2 or ϕ
Image(R, M), function	Yields the image of R under map M ($\subseteq$ E x E)

Temporal primitives establish relationships among fluents and contexts. Eval yields the value of F in X if F is a property of an object. If F is a set, it yields the members. If F is a function, it executes it and returns the value, if any, returned by the function. The context S is the context for the evaluation, e.g. from the <context root> and local dynamic context.

Table 6-4 Temporal Primitives

Primitive	**Characteristics**
Holds(S,F), Predicate	Fluent F holds in context S
Eval(F,S), Operator	Computes the value of F in context S
[Si < Sj](e_0, b) Predicate	Context Si precedes Sj b-near point e_0 in E

As objects such as cell towers and other radios enter and leave the local environment of the cognitive PDA. The axioms reflect this using ***Appeared*** and ***Disappeared***. Location and position refer to self, fixed, or last known absolute spatial characteristics of the objects.

Table 6-5 Physical Primitives and Functions

Primitive	**Characteristic*s***
Exists(O), function	Fluent of the point set occupied by O at E_X
Present(O), function	Fluent of the point set occupied by O $\cap$ S
Location (O), function	Maps shape(O) b-near to e_0 on O as a (Latitude, Longitude) pair or GPS vector
Position (O), function	Fluent that orients O at e_0 as (x, y), (x, y, z) or sεS
Appeared(O,X), function	Object O appeared in context X
Disappeared(O,X), function	Object O disappeared in context X

6.4.4 Executable Models

Executable code is a formal model. Bindable expressions from computer languages that are incorporated into RKRL are designated using the name of the language as the <model>. An executable expression performs according to the language specification. Expressions in RKRL from interpreted languages like LISP perform the task indicated by the context and/or body when bound and invoked. In RKRL 0.1, Excel was the host language platform. Therefore, expressions like "=Weight 3 Model * Tap 3 Model + Weight 2 Model * Tap 2 Model + Weight 1 Model * Tap 1 Model" compute the values indicated when installed in a cell in an Excel worksheet. In this sense, Excel is a dynamically interpreted language suitable for building self-extending systems. The <model> family "Excel" designates the body as such an expression. An Excel macro can create a statement using string operations and insert it into a worksheet cell where it is immediately interpreted. This permits a macro to extend an Excel spreadsheet based on data in the spreadsheet. A macro cannot create new macros that way, so its extensibility does not extend to the automatic creation of new meta-level primitives.

As an object-oriented language, Java's data structures are readily extended, e.g. with the srModels of CR1. Java strings are not as readily attached to a variable and Evaluated as programs (methods) from within a Java program as LISP, or even Rexx or Perl. Java was used because of the large number of software packages, including freeware, that facilitated the rapid prototype implementation on a short schedule. The WebL Java [85] dialect is simpler and easier to extend than Java. The Java dialect JPython [182] also permits user scripting, which is a limited form of extensibility. In addition, both WebL and JPython are dynamically typed, alleviating many of the problems with attempting to extend Java directly at run time.

Machine learning in the form of acquiring new srModels does not require such dynamic-extensibility. Machine learning sometimes includes acquiring new skills. For example, a core machine learning program could construct a program to control a new effector by synthesizing an explanation of its interface (API) into a program that calls the effector. This requires a run-time extensible language with persistence (so the new methods defined in a session are not lost). Web interest in mobile agents seems to be creating renewed momentum for auto-extensible languages. The language needs for machine learning fly dramatically in the face of "good programming practice." Writing self-modifying code has been practiced since at least the IBM

709 era, but has been in disfavor as a programming style in the past. The renewed emphasis on machine learning forces one to rethink this predisposition. It might be better to define characteristics of "good" self-modifying code (e.g. provable stability, politeness in asking for permission before modifying itself, etc.) to avoid "bad" self modifying code.

6.5 Micro-worlds of Space, Time, Spectrum, and Users

The space, time, and users micro-worlds implement the axiomatic strategy presented above. The environment-aware use case requires reasoning about radio propagation at the physical layer and about means for sustaining services on the higher layers of the protocol stack. RKRL therefore includes a spatial inference hierarchy organized into layers or "spatial planes" for the above scenarios. The Global plane supports international travel, the Regional plane for shorter trips, the Metropolitan plane for urban areas, and a Local plane for the immediate vicinity. The knowledge structures are useable both horizontally and vertically. Horizontal reasoning refers to inferences within a specific plane, such as matching an international travel itinerary to a list of world cities. This reasoning generally occurs within a limited context (e.g. planning a trip or on travel). Vertical reasoning refers to the use of multiple levels of the spatial hierarchy, such as using knowledge of the structure of a Metropolitan area to access the appropriate wireless band and mode. Sometimes a mix is required as in the following example.

The Time micro-world defines temporal primitives from nanoseconds to millennia. The primitives (in alphabetical order) are: days, Decades, Epoch, Hours, Instant, Interval, microseconds, Millennia, milliseconds, Minutes, months, nanoseconds, Seconds, Session, years. In addition, temporal epochs have subsets corresponding to atomic stimuli, phrases, dialogs and scenes. Atomic stimuli are parsable tokens. The associated temporal epoch is the time required to sense a parsable token. Phrases are sequences of parsable token that occur between phrase markers. Phrase markers include time epochs of a specified duration and changes of context including change of speaker. Scenes are identified by communications context. This set of temporal subsets is tailored to the specific needs of the wireless PDA use-cases of the previous chapter.

The Spectrum micro-world contains nearly 300 frames describing radio spectrum allocations. There is also an embedded model of potentially pooled spectrum that was used in

the use case above. As cognitive radio is implemented, this micro-world must include the detailed characteristics of each band within which the radio will operate. This micro-world includes preliminary templates of signals accepted for operation in each band. This includes the usual air interface characteristics of allocation, RF channelization, modulation, power envelopes, etc.

The User micro-world defines characteristics of users, including Sex, Name, and Home. Capabilities of users are also defined: Record (as in, to write something down), Remember, Sell, Buy, Hear, See, Relieve, Speak, Listen, Walk, Sleep, Eat, and Plan. In the development of CR1, the user micro-world was expanded to include dialogs from foreign language phrase books. CR1 was trained on these dialogs using the material from RKRL 0.3.

6.6 The Spatial Micro-worlds

The ontological micro-worlds provide the axiomatic treatment necessary to define concepts unambiguously and to support the automatic generation of proofs (e.g. of the viability of a plan). The spatial micro-worlds are those ontological micro-worlds with the data structures and related domain knowledge necessary to accomplish mobile wireless tasks using spatial reasoning. These micro-worlds consist of the Global plane, Satcom, the Regional plane, the Metropolitan) plane, and the Local micro-world, which defines local, immediate, and fine-scale spatial structures.

6.6.1 Navigating Spatial Planes

Since the spatial calculus is defined over sets, a spatial plane is a representation of a subset of space. The default test for inclusion in a spatial region is the intersection of the radio's present-location with the set (or a subset) of the region in question. For example, the region Europe includes an explicit list of cities like Stockholm. Therefore, if the set that represents present-location ("Here" in CR1) includes Stockholm, the intersection of the present-location set with the cities of Europe contains at lest the point "Stockholm". If Here includes a latitude-longitude tuple, then the computational model of set intersection for that element is a search of latitude-longitude boxes using the metrics from R. If the intersection of present-location with all the spatial planes is empty, then the cognitive radio does not know where it is located.

But location is defined on many levels including distance from the nearest cell tower.

Thus, spatial reasoning micro-worlds include the location of fixed transmission facilities. A cognitive radio could employ its KQML-ask skill to obtain this information from cognitive infrastructure. Regions are included in each other to the degree that their sets overlap. Thus, cognitive radio's internal characterization of its location is the set of all spatial sets that intersect with items independently placed in its present-location set. Its GPS receiver, for example, provides an independent input to this set. A network's reply to a KQML query would also be independent input. The inference drawn about whether the radio is indoors or outdoors from its environment sensors is also an independent input. This representation mechanism supports reasoning about the mutual consistency of members of present-location.

6.6.2 Global Plane

Figure 6-13 shows the top level of the physical-world inference hierarchy. This is the global plane in which telecommunications patters that are global in scope are represented. This can include global demographics such as population, global connections, global connectivity statistics, and global calling patterns. Its primary purpose is to represent deep knowledge about a specific user, such as the owner of a mobile PDA. If that user is in the middle of a trip around Europe, for example, his travel itinerary is mined for information useful to both the PDA and to the networks in which it finds itself during the trip.

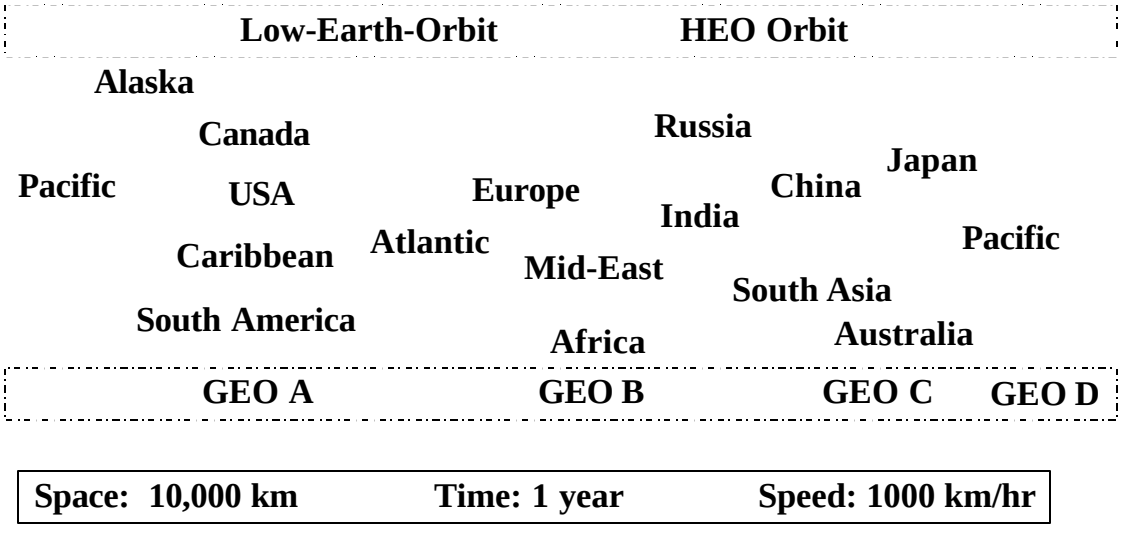


Figure 6-13 Global Plane of the Inference Hierarchy

The global plane micro-world includes the countries of the world, the oceanic areas used

for telecommunications and the satellite systems that connect more than one country at a time. This space has a characteristic distance of about 10,000 km. The characteristic distance is a measure of the average spread in distance among objects at this level of the hierarchy. This plane has a characteristic time of about a year. That means that its properties should be updated about once a year. In addition, significant motion in a domain requires a rate of speed that can transit a substantial fraction (>10%) of the characteristic distance in an hour. Such a traveler would be traveling at 1000 km/hr, e.g. in an aircraft.

In addition to a set of objects with aggregate characteristics, each plane has additional properties including specific interconnection mechanisms, movement patterns and other mechanisms as defined above. These properties include the following:

1. Characteristic Interconnect: Fiber trunks, Satcom
2. Characteristic Travel: Air (rail, ship)
3. Temporal Patterns: Annual (vacation, business events), demographics, day-night zones
4. Space-time: User Itinerary filed with network
5. Information: Location, telephone #, Internet-address, mobility, path length, delay, data rate, QoS, traffic density

Items in parenthesis such as (rail, ship) may be relevant to the topic but would generally be secondary in nature. The space-time mechanism captures the ways in which relationships among space and time are aggregated in this domain. The information aspect indicates the factors that most effect access to information or availability of information. At the global level, one's location in the world has the first order impact on access to information infrastructure. Even with the forthcoming operational status of Iridium, access to ISDN, video teleconferencing and high speed internet services will be driven by location. In addition, one must have a known address to be reachable. And the characteristics of the information pathway must be capable of supporting the information service.

6.6.3 Regional Plane

The next level down the RKRL spatial inference hierarchy is the regional plane. Each

member in the global-plane (e.g. Europe) has RKRL frames for its own representation set that defines its structure. Figure 6-14 illustrates part of the regional plane for the European Community. Each plane is faithful to real-world data to the degree necessary for cognitive radio's goals. When transiting through the airport, one does not need much detail. When arriving for an extended stay, greater detail is appropriate. In addition, the degree of error on each plane is represented by open balls in the applicable topological space.

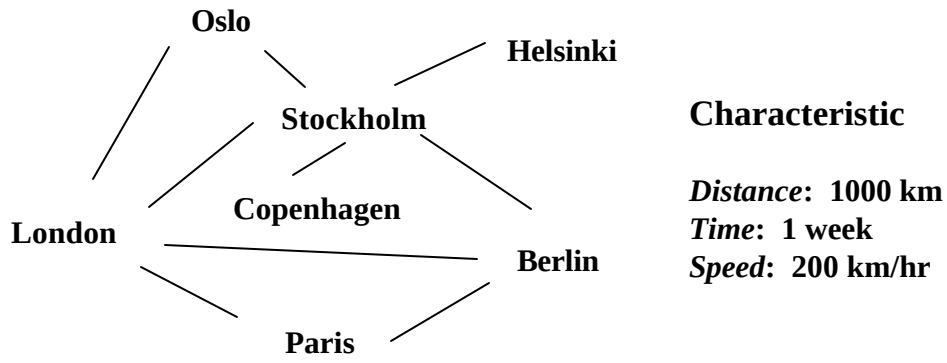


Figure 6-14 Regional Plane Consists of Major Urban Areas in a Global Region (e.g. EC)

The additional properties of this region include:

1. Interconnect: Fiber trunks, Satcom (Terrestrial microwave)
2. Travel: Commute by Air (rail, ship, automobile)
3. Temporal Patterns: Weekly (annual, daily), seasonal
4. Space-time: Itinerary, commuting habits, day-night boundary
5. Information: same as global level plus constraints imposed by the nature of geopolitical boundaries such as national borders; physical barriers such as mountain ranges; etc.

6.6.4 Metropolitan Plane

The next level of the cognitive radio hierarchy is the metropolitan region illustrated in Figure 6-15. Metropolitan areas generally contain the greatest intensity of telecommunications infrastructure. Clearly, regions defined at this level may not be metropolitan areas demographically. A large wilderness parkland, for example, could be on the metropolitan plane

of the inference hierarchy. Stockholm located within 100 km of the hub could be of relevance to service-delivery tasks of cognitive radio, depending on the location of the user's home, depending on vacation habits, etc. In addition, commuters have a daily pattern that involves movement among these regions, typically by rail or automobile. RKRL's representation sets are defined to capture information that is relevant to services delivery, while not creating a computational burden. A protocol for updating a cognitive PDA's local store of this level of knowledge could be built on RKRL's KQML facility.

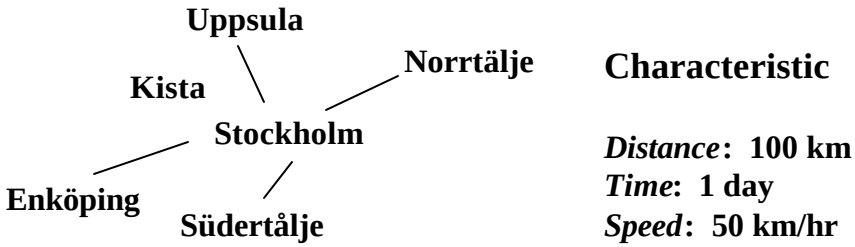


Figure 6-15 Metropolitan Area Plane Defines Adjacent Geography and Infrastructure

Other significant properties of this level of the inference hierarchy include:

1. Interconnect: Wireless, Fiber trunks, (Satcom) Propagation: Coverage is the primary issue
2. Travel: Commute by Rail, auto, (air, ship)
3. Temporal Patterns: Daily, (weekly, annual, seasonal)
4. Space-time: Commuting habits, to-do list
5. Information: Wide area access, best service provider

At this level of the hierarchy, the extent of wireless coverage becomes important. In addition, temporal patterns shift to a daily cycle driven by commuting and leisure pursuits, depending on the day of the week and the season. A commuter's normal pattern will be shaped by a notional "To-Do" list, which may cause significant variations in the space-time pattern of demand offered for wireless. Visits to clients, luncheon engagements, etc. can shape the needs of wireless power-users such as corporate executives. Since power-users may be early adopters of the new level of services contemplated in this research, their patterns provide relevant use case

scenarios in which to test cognitive radio concepts.

6.6.5 Local Plane

Continuing down the initial inference hierarchy, one encounters the local area plane illustrated in Figure 6-16. The local area is the region that is within a 5-10 minute walk, subway, or automobile ride. A subscriber's location will typically vary over this area throughout the course of a day. In the example of the figure, a visitor to the SAS Royal Viking hotel in downtown Stockholm might take a walk around town. The Åhlens and NK department stores are located within a few hundred meters of the hotel and of each other. This is adjacent to the large green area of the Kungstradgarden, a parklike walkway to the river. The vicinity of the Opera is adjacent to the old city of Gamla Stan in which there are a number of restaurants and shops. The bridge connecting Gamla Stan via the Centralbron to the Central Station rail terminus provides convenient access back to the Royal Viking. These key places in town are part of the local area that constitute important elements of the spatial domain for a user visiting this part of Stockholm.

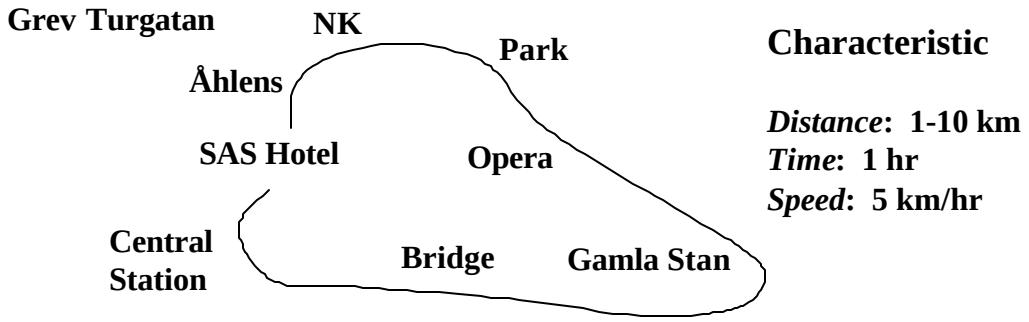


Figure 6-16 Local Area Plane of the Inference Hierarchy

Additional aspects of this radio access space include:

1. Interconnect: Wireless, Cordless, Satmobile (Propagation: Reflection, scattering, multipath; preferred Bands: VHF, UHF, (EHF, HF)).
2. Travel: Foot, taxi, train
3. Temporal Patterns: Daily, (weekly, seasonal)
4. Space-time: To-do list, meals

5. Information: Local access, local coverage/ gaps

The local area plane is the logical level of abstraction for a next-generation level of depth in defining wireless access. The higher levels support a travel scenario, but this level supports routine daily use of wireless. Fade zones, areas of constructive interference and overlap among coverage of service providers may be captured by a cognitive radio by cataloging signal fluents across the dimensions of interconnect, space-time and information access.

6.6.6 Immediate Area

But fourth-generation capability could also reach to the next level of the hierarchy. This is the immediate area plane illustrated in Figure 6-17. Although the local area can be described using two or 2 ½ dimensional models, (x, y, and height in selected places), the immediate area requires the full three dimensional treatment. Sensors of an environmentally aware radio characterize this plane as it continuously changes to reflect the immediate vicinity of the owner of the mobile information appliance. The characteristic distance of this area ranges from one to a hundred meters. The characteristic time scale is on the order of seconds to minutes.

Characteristic

***Distance*: 100 m**
***Time*: 1 minute**
***Speed*: 1 m/sec**

Figure 6-17 Immediate Area Plane Characterizes Propagation in Three Dimensions

In addition, the information focus has changed as suggested by the following characteristics:

1. Interconnect: Voice, Wireless, Cordless

   Propagation: Vehicular reflection, multipath

   Preferred Bands: VHF, UHF, (EHF, HF)

2. Travel: Foot, (taxi, train refer to inside of vehicle)

3. Temporal Patterns: Hourly/ momentary action needs

4. Space-time: Use of artifacts, dead reckoning navigation

5. Information: Nearest infrastructure access point

Travel in this plane is limited to walking or movement within vehicles which themselves may be moving. On the global plane, the user is one of millions of others represented from a statistical perspective. A unique itinerary in the global plane allows the network to plan and deliver customized support for the user. But, that plane itself is itself not user-centric. The immediate area plane, however, is the user-centric plane. It is best described by a coordinate system that moves with the user, reflecting his immediate environment. Its creation will require a mix of models. The distribution of data about this plane would certainly place significant demands for bandwidth on conventional cellular infrastructure. One wonders whether the cost would be worth the benefit. A local RF LAN, however, could deliver large volumes of data about one's immediate area at low cost.

6.6.7 Fine Scale

A still deeper level, the fine scale plane, is probably beyond the descriptive capabilities of a cost-effective network. This plane consists of the objects within 10 meters of the user as suggested in Figure 6-18. Assume that a cognitive radio PDA is carried or worn by the owner. It may interact with other objects in the environment such as the owner's personal computer (PC), e.g. using Bluetooth, or an Infrared port (IRDA).

Characteristic
***Distance*: 10 m**
***Time*: 1 msec**
***Speed*: .1 m/s**

Figure 6-18 The Fine Scale Plane Characterizes Each Significant Object

Propagation effects in the fine scale plane are those that change over fractions of a wavelength. This plane has the following additional characteristics:

1. Interconnect: Physical Contact

   Propagation: Reflections from body parts, walls, furniture, appliances

   Preferred Bands: Infrared, optical, EHF, (SHF)

2. Travel: Movement of body, artifacts

3. Temporal Patterns: Segment of momentary motion

4. Space-time: Fine scale effects

5. Information: Very low power local exchange possible

Space-time patterns could be induced from the unit's own knowledge of its own location plus measurements continuously made by the unit. One benefit of knowledge on this level is the design of very low-power high-data-rate exchanges among information appliances. The hand held unit could periodically probe the RF bands to determine if there is a computer or other information appliance in range. Cognitive radio could employ its fine-scale to enhance local connectivity.

This concludes the definition of the spatial inference hierarchy.

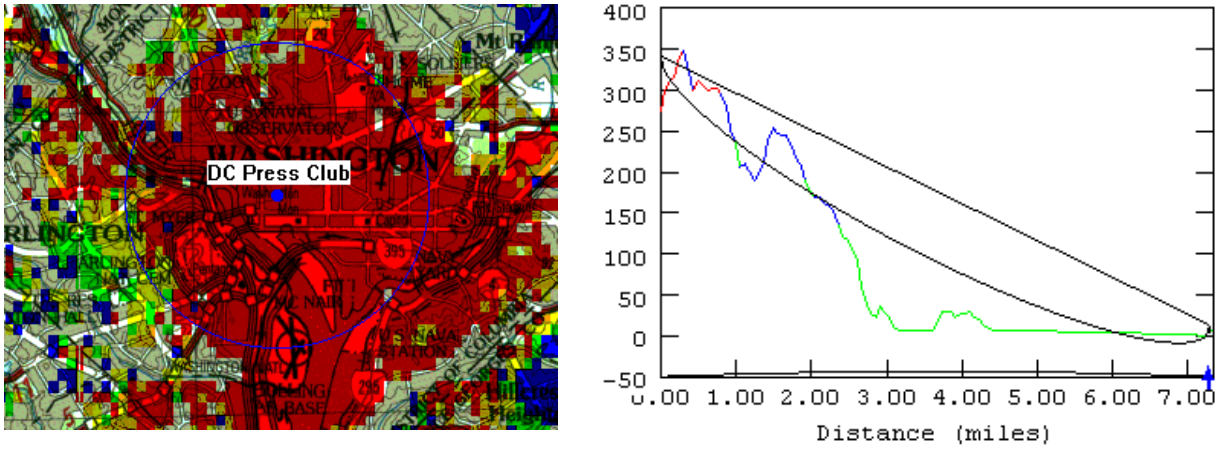
6.7 The Generic Radio Micro-worlds

The Generic Radio micro-worlds define those aspects of radio that are generally applicable to terrestrial mobile wireless. These micro-worlds include Propagation, Architecture, Functions, Internal, and Hardware

6.7.1 Propagation

The Propagation micro-world consists of 33 frames that identify the interfaces to propagation models. This micro-world describes the Walfisch model to provide an exemplar. The spatial scale of these models is the local to immediate planes of the spatial hierarchy. The functional interfaces expressed in RKRL are the (fact of a) terrain model, the type of RF parameter estimated (e.g. received signal strength), and the fidelity of the estimate. In addition, RKRL requires a resource estimate. This could be an elapsed time estimate for typical and worst-case situations on the target machine. These axioms and interfaces permit one to integrate an existing propagation model as an inference component in a cognitive PDA as illustrated in

Figure 6-19 [183, 184].



RFCAD Propagation Map and Contour Profile, © Biby Engineering Services

Figure 6-19 Illustrative Contributions of Propagation Modeling Components

The best models require a three dimensional map of the local area that is accurate to 1 meter in three dimensions. Typical propagation models at this scale are accurate to about 2 to 5 dB [185]. Embedding such models into the propagation micro-world can predict the extent of deep fades given the location of nearby cell towers and buildings. Cognitive radio could use this information to avoid offering a long file to a network in an area where it has low probability of fast error-free transfer on a given air interface.

6.7.2 Internal Plane

The Internal micro-world describes the components of a software radio. RKRL 0.3 includes over 100 frames in the Internal micro-world. This micro-world enumerates Hardware, Software, Modem, Demod, Equalizer, and Memory as subordinated micro-worlds. The description includes black-box capabilities and parameters for Antenna, Antenna Arrays, Baseband Processor, IF Processor, Modem, ODP software for distributed processing, and RF Conversion. This plane is illustrated in Figure 6-20. It contains or points to all the sets that represent the internal states of the unit itself. This plane allows the unit to perform self-referential reasoning. Such reasoning is necessary, for example, for the unit to determine whether a download of a new personality or capability would be beneficial or detrimental. In a single-mode network-centered world, handsets do not need such capabilities. In a multi-band

multi-mode terminal-centric future, such reasoning capabilities will be essential to protect the PDA from unintentional injury by the networks. This plane contains a simple functional model of the radio unit such as the functional model proposed by the SDR forum. In addition, it contains configuration data on the hardware, software, and firmware as illustrated in the figure. This micro-world may refers to the architecture, hardware, and software micro-worlds for additional detail. These items could include analog parts, FPGA's, DSP's, buses, and general purpose computers. Each is characterized at least in terms of function, interfaces, processing capacity and memory.

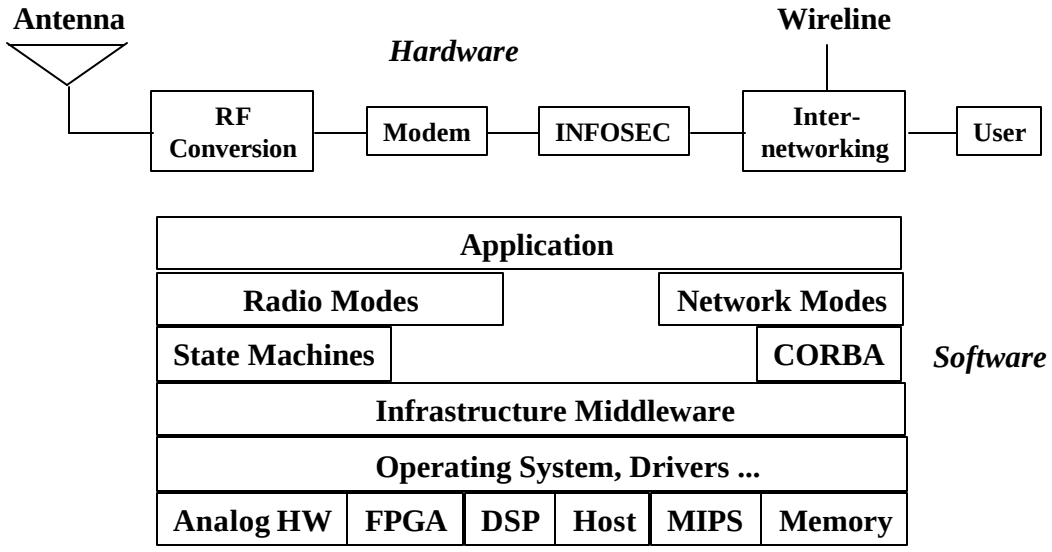


Figure 6-20 Internal Plane

This plane also has information access concepts including the following:

1. Interconnect: Electronic and logical inside of node
2. Travel: Movement of data in streams, packets, CPU
3. Temporal Patterns: Isochronous streams, instruction cycles
4. Space-time: Data capacity of info path/ protocol/ reliability
5. Information: On which processor is the data/program hosted?

Interconnect now refers to the process of joining items inside the radio to each other. The antenna is connected to RF conversion, etc. One "Interconnect" is defined in a set-theoretic way, the facilities apply as well to the interconnection of an antenna and an RF converter as to

interconnecting the US with Europe. Travel at this level of abstraction now concerns movement of data. Data includes applications data such as streams of digitized voice. It also includes instructions packaged as data loads and comprising personalities or capabilities for the radio node. Space-time now refers to the distribution of data on processors. Information access is determined by the location of the data on specific processor or memory components.

6.7.3 Architecture

The Architecture micro-world consists of over 90 frames that define the generic architecture of the software radio. It identifies functions and components, including the standard reference framework of the SDR Forum. The set frame for this micro-world is as follows:

(Channel Coding and Decoding , Channel Set , CORBA Middleware , Evolution Support , IDL , INFOSEC , Joint Control , Modules , OE , Personality , Service and Nework Support , Source Coding , Source Decoding , Source Set , Views ,)

6.7.4 Functions

The Function micro-world defines the radio functions of Access, Networks, Access Points, Air Interfaces, Components, Core Networks, Global Services, and Mobile Units. It includes pointers to the Radio Propagation Environment. Its set frame is:

(Access Networks , Access Points , Air Interface , Components , Core Networks , Global Services , Mobile Units , Radio Propagation Environment ,)

6.7.5 Hardware

The Hardware micro-world defines the hardware concepts of Hardware, Module, Availability, Red (side of an INFOSEC device) Black side, IDEF1, Data Model, and Maintenance. It also defines specific hardware modules including Chassis, Exciter, Phone, Receiver, and Transmitter.

6.8 The Software Micro-worlds

The Software micro-worlds define software and specific software tools needed for software

radio including CORBA, UML, ODP, and MPI.

6.8.1 Software

The Software micro-world defines the software concepts, not software configuration items. These include Applications, Framework, Operating Environment, Operating System, POSIX, Resource, and Services. This micro-world also has pointers to CORBA, IDL, KIF, KQML, ODP, and UML. Theses definitions provide the links necessary to expand the knowledge (e.g. via web queries) as the need arises.

6.8.2 CORBA

The CORBA micro-world defines the core concepts of the Common Object Request Broker Architecture. This includes the basic concepts (e.g. ORB, IDL) plus the software radio aspects of a reuse Factory, Core Services, Core Applications, and Base software radio interfaces. The integration of the SDR Forum's domain manager into this micro-world would be a useful extension to the present research.

6.8.3 UML

The Unified Modeling Language (UML) micro-world defines UML concepts. These include Relationship, Active, Class, Source, Target Class, Inheritance, Child, Parent, Multiplicity, Association, and Uses. The purpose of this micro-world is to provide a starting point for the future definition of a direct interface between RKRL and UML.

6.8.4 ODP

The ODP micro-world defines the core of X.900 capabilities needed to reason about control of distributed processing. This includes to suffer-delay and to fail. Basic sets include Information and Distributed Processing.

6.8.5 MPI

The MPI micro-world defines software processes and process groups using the Message Passing Interface (MPI) process control language.

6.9 The Wireless Function Micro-worlds

The wireless functions micro-worlds define air interfaces, providing specific details on modem, demodulation, equalization, and memory.

6.9.1 Air Interfaces

The Air Interface micro-world consists of over 250 frames that define general knowledge of air interfaces including bitstreams, channel states, modulation, and power control. It provides the core from which users could evolve a comprehensive taxonomy of all known and proposed air interfaces. This supports the autonomous development of new interfaces in terms of features of existing or proposed interfaces. Selected subsets of this full taxonomy would be downloaded to mobile units to assist them in conducting dialogs with new networks during global roaming.

As a minimum, a production-quality micro-world would define CDMA, GSM, DECT, PCS-1800, PCS-1900, UWC-136, and a generic air interface. Additional concepts of this micro-world include Error Control, Framing, Multiple Access, Multiplexing, Bit Rate (Rb), Security, Source Coding, and Spread Spectrum. A Generic air interface is defined to include Carrier Bandwidth, Frame Rate, Handoff, Handover, Power, RF Channels, Sectorization, Services Access Bandwidth (Wa), Baseband bandwidth (Wb), RF Channel bandwidth (Wc), and IF Bandwidth (Wi).

6.9.2 Modem

The Modem micro-world contains 50 frames that define the modem functions of modulation and demodulation. Examples of modulators include 16 QAM modulator, 64 QAM modulator, 8 PSK modulator, BPSK modulator, and QPSK modulator. Concepts for modems consist of bits per symbol, bitstreams, carrier frequency, IF waveform, baseband, IF/RF carrier, sample instant, and symbol duration. Types of modulation defined include AM, FM, PSK, FSK, MSK, and QAM. The parameters for control of the modems are patterned after [154].

6.9.3 Demod

The Demod micro-world defines the internal functional building blocks of a demodulator including bit decision, carrier detection, carrier tracking, despreader, equalizer, filter, squelch, symbol decision, symbol enhancement, symbol estimator, timing recovery, and timing reference.

6.9.4 Equalizer

The Equalizer micro-world defines a generic equalizer. Its functions includes the delay line, weight matrix, and weighted result. This micro-world includes a three-tap equalizer model embedded into the RKRL statements. The RKRL statements from Version 0.1 are as follows.

Handle	***Model***	***Body***	***Context***
Demodulator	Contains	Equalizer	Modem/Internal…
Equalizer	Contains	Delay Line	Demodulator/Modem…
Delay Line	Contains	Number of Taps	Equalizer/…
Delay Line	Contains	Tap Vector	Equalizer/…
Delay Line	<Range>	Tap Vector	Equalizer/…
Delay Line	Contains	Taps	Equalizer/…
Taps	Contains	Tap 1	Delay-Line/Equalizer/…
Tap 1	Numerical value	0.995	Taps/Delay-Line/Equalizer/…

The complete extended context is not shown in the frames. Each tap is defined by similar frames. The "Numerical-value" of Tap-1 is the value currently in use by the software radio. The parallelism of the pattern Tap-1, Tap-2, Tap-3, supports automatic extensions of an air interface. From such a structure, a genetic algorithm could hypothesize additional taps, and RKRL readily supports the implementation of the hypothesized new equalizer. In addition, other statements show that each is linked to the other by a unit delay element. The output of the tapped delay line is defined through additional RKRL statements as follows.

Handle	***Model***	***Body***	***Context***
Equalizer	Output	Weighted Result	Demodulator/…
Weighted Result	Numerical value	1.1209227	Equalizer/…
Weighted Result	Definition	Sum of Products of tap values times weights	

The complete model of the three-tap equalizer requires about 50 RKRL statements.

6.9.5 Memory

The Memory micro-world consists of a model of memory allocation for a demodulator that includes a separate equalizer memory allocation.

6.10 The Protocol Micro-worlds

The Protocol Stack micro-worlds consist of Protocol, Physical, and Data Link.

6.10.1 Protocol

The Protocol micro-world consists of about 300 frames that define the basics of network protocols. This includes establishing connections, maintaining connections, use of Protocol, and termination of connections. The ISO protocol stack is defined to consist of an Applications layer, a Data Link layer, a Network layer, a a Physical layer, a Presentation layer, a Session layer, and a Transport layer. The hierarchical relationship among these layers is also defined. This micro-world also defines Platforms, PDU (protocol data unit), and Service Facilities. Other aspects defined in this micro-world include E-Care, IS-683A, OTA (over the air), OTAMD (mobile diagnostics), OTAPA (parameter administration), OTASD (software download), OTASP (service provisioning), general data link layer Processes, TCP, and the WAP (Wireless Applications Protocol).

6.10.2 Physical

The Physical micro-world provides a more detailed definition of this layer of the protocol stack. It includes procedural access to the physical medium, functional access to the physical medium, electrical access to the physical medium, and mechanical access to physical medium.

6.10.3 Data Link

The Data Link micro-world define concepts and models for this level of the protocol stack. It includes error control, Error detection, FEC, flow control, frames, Header, Packet Number, Packetize, Payload, Sequence Number, synchronization, Timing, and Training Data.

6.11 The Network Micro-worlds

The Network micro-worlds consist of Network, Cellular, Segmentation, and Messages.

6.11.1 Network

The Network micro-world is one of the largest at over 400 frames. It defines Air Interface and radio Functions from a network perspective. It also defines ATM (and its AALs), cellular,

connection, destination, SAP (service access point), encapsulation, re-assembly, router, segment, segmentation, and simulation. traffic classes include CBR (constant bit rate), ABR (available bit rate), and VBR (variable bit rate). Subclasses of VBR include timed VBR, connection-oriented VBR, rt-VBR, nr-VBR, connectionless VBR. A generic traffic contract is defined in terms of QoS Parameters. This includes ATM-oriented definitions of CDVT (cell delay-variation tolerance), CLR (cell loss rate), Feedback, maximum CTD (cell delay tolerance), MBS (maximum burst size), MCR (minimum cell rate), PCR (peak cell rate) Peak-to-peak CDV (cell delay variance), SCR (sustained cell rate), and other traffic parameters.

6.11.2 Segmentation

Segmentation consists of an embedded computational model of the segmentation of a message into packets, affixing header and trailer, etc. This model is an example of an independent model of the function of a typical layer of the protocol stack. Using such a model, cognitive radio can independently determine whether the a segmentation module is functioning according as it should. Using such a model, cognitive radio will also be able to incorporate segmentation into automatically defined protocols.

6.11.3 Messages

The Messages micro-world defines message handling functions. These include control, format, generate, get, next, propagate, receive, respond, send, and transmit. In addition the concepts defined in this micro-world point to the other protocol micro-worlds.

6.12 Mechanisms for Extending RKRL

RKRL may be extended using the ***expand()*** operator on a ***contains*** frame.

Expand (Frame $_i$) = Frame $_{i+1}$.

In the resulting frame, the new handle names the subset to be expanded. This may consist of the enumeration of subsets or new point sets using by using the contains model in the new frame. Alternatively, the new frame may provide a characterization of that subset using any other models.

The define() operator creates frames needed to characterize the body of the argument as a

topological map, as follows:

Define (<Handle><Model><Body><Context><Resources>) =

1. <Body>Domain <Domain subset> …
2. <Body> Range <Range subset> …
3. <Body> Process <Process Specification>

These three frames require the body of the argument frame to be specified in terms of the point-sets on which it operates (domain and range) and the transformation that it makes. This transformation may be encapsulated as the name of a software module that accomplishes the transformation. Alternatively, it may be expressed as CORBA IDL that shows how to request the transformation from an object request broker.

Auxiliary functions such as find(), sort(), etc are needed to manipulate frames to assure that new frames are integrated into the existing RKRL frames effectively.

7 The CR1 Prototype

The purpose of the CR1 prototype is to refine the initial framework of cognitive radio. This required the integration of machine learning, natural language processing, and radio architecture. Radio domain knowledge was organized into reinforced hierarchical sequences. These internal data structures that are processable across a continuum from radio protocols to natural language. In addition, these data structures facilitate machine learning. CR1 implements case-based learning, but the architecture derived from CR1 supports a range of contemporary machine learning techniques. The development of CR1 led to the definition of the cognitive radio architecture discussed in the following chapter. This chapter describes the CR1 rapid prototype in terms of its operation in a simulated environment, its software architecture, the way it performs radio control tasks, and performance metrics. All of the capabilities described have been simulated in Java. The result is a simulated PDA that could be implemented with technology available within five to ten years. The thesis addresses research issues and not implementation issues. The primary research issue addressed is the organization of radio domain knowledge into data structures processable in real-time that integrate machine learning and natural language processing technology into software radio.

7.1 Illustrative Scenarios

CR1 is envisioned as a hand-held PDA. The PDA face, illustrated in Figure 7-1, represents sensors and effectors. These address time, space, radio, and user interactions. Effector displays are editable as user input. Axiomatic time uses Java facilities to determine the Year, Month, Day, Hours, Minutes, and Seconds of the global "nowField." The Now display area is both a sensor display and an effector. If the user wants to interact with the PDA about the future (e.g. planning a trip), the user enters the time of interest in the Now field. The PDA can tell that this is hypothetical time because it is not identical to the Now field.

The "Lat" and "Lon" fields display GPS latitude and longitude coordinates when available. The PDA's Azimuth heading is also displayed from GPS with inertial updates (e.g. from a MEMS sensor). Local grid references provided by wireless nodes such as the IR badge system

at KTH are registered in the X and Y fields. Since these are editable, a user could tell the PDA "The Coke Machine is here" by editing the X and Y fields. Place is displayed by the PDA from its interpretation of Lat, Lon, X, and Y sensors. The rapid prototype does not use altitude, speed, or velocity, although such parameters could help set communications context. The subset of parameters chosen addresses the research issues.

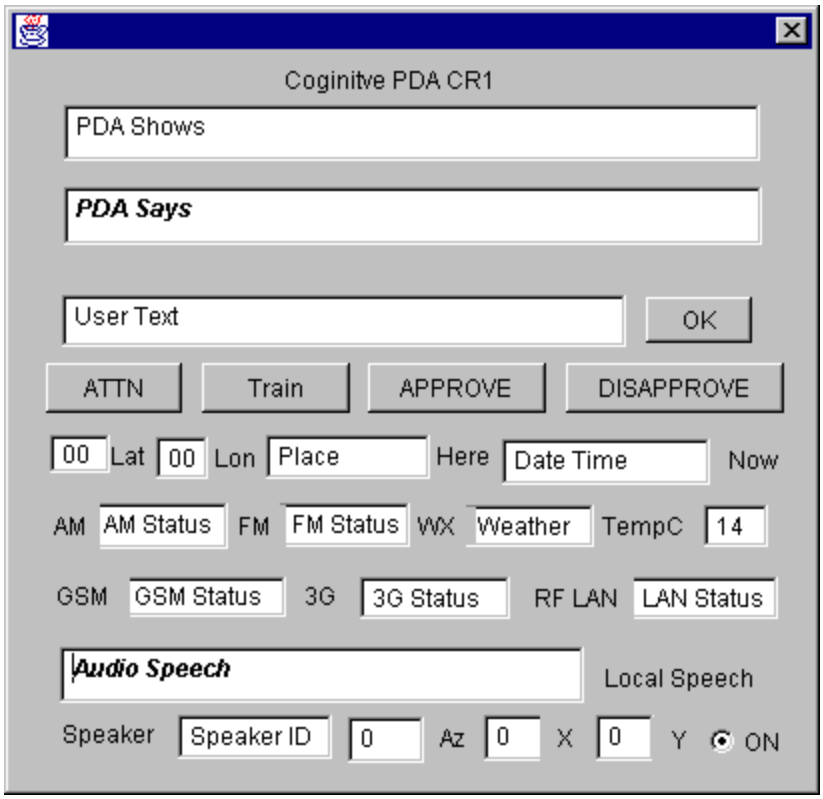


Figure 7-1 The CR1 Prototype PDA

Keyboard inputs are supplied in the User Text area. The radio resources are displayed in a set of radio-channel status displays. These consist of RF-LAN, PCS, GSM, AM, FM, and weather broadcast (WX, e.g. the US National Oceanographic and Atmospheric Administration - NOAA - weather channels). Temperature shows the average of its Temperature-Front and Temperature-Back sensors that would be used to determine if the PDA is being held or not. The Audio Speech area represents notional spoken inputs in textual form. It is called Speech-Front because it reflects the user input. A Speech-Back sensor (not simulated) would pick up background noise for active cancellation. Speech-Back could also recognize aspects of the environment (e.g. quiet, discernable conversation, noisy, etc.). SpeakerID tells the user who the

PDA thinks is talking. This can include the speaker's name or a generic identity (e.g. Speaker1).

The user can express feedback in a way that is more heavily reinforced than speech or text using the TRAIN button, ATTN (attention) button, APPROVE button, and/or DISAPPROVE button. The attention button resets the lower levels of the PDA's internal states so that the user's inputs are not cluttered by its monitoring of the environment. As long as ATTN is depressed, the PDA responds only to changes in user inputs, not to the background (e.g. speech, incoming email, RF states). These could be processed and queued, but not enter the dialog. Not shown is the Light Level sensor, which helps the PDA determine whether it is indoors or outside.

The PDA's effectors include the X, Y, and time displays; all of the radio-related fields; a text display area; and a speech synthesizer. The "Shows" field contains the PDA's textual responses. In addition, the user can edit this field to advise the PDA of what it should have said or should say in a given context. The "Says" field shows in textual form what the PDA would provide to its speech synthesizer.

The identification of a user's communications context and the definition of wireless use patterns in a way that enhances traffic shaping benefit from machine learning. The mutual associations of such sensory stimuli mapped to user communications context (e.g. Table 7-1).

Table 7-1 User Communications Contexts and Sensors

Context	**Illustrative Sensory Stimuli**
Home (Outside)	GPS (250m), Light(Day →Bright; Night →Dark), No LAN
Home(Inside)	No GPS, Light (Dim), Home LAN, CT Home #, Recent Home(Outside)
Neighborhood	Strong GPS(Home), weak CT signals
Spectator	Neighborhood/ Soccer-related Speech
Commuting (Automobile)	Strong GSM, Weekday, 6-7AM, Auto Noise in Acoustic Background

CT= Cordless Telephone

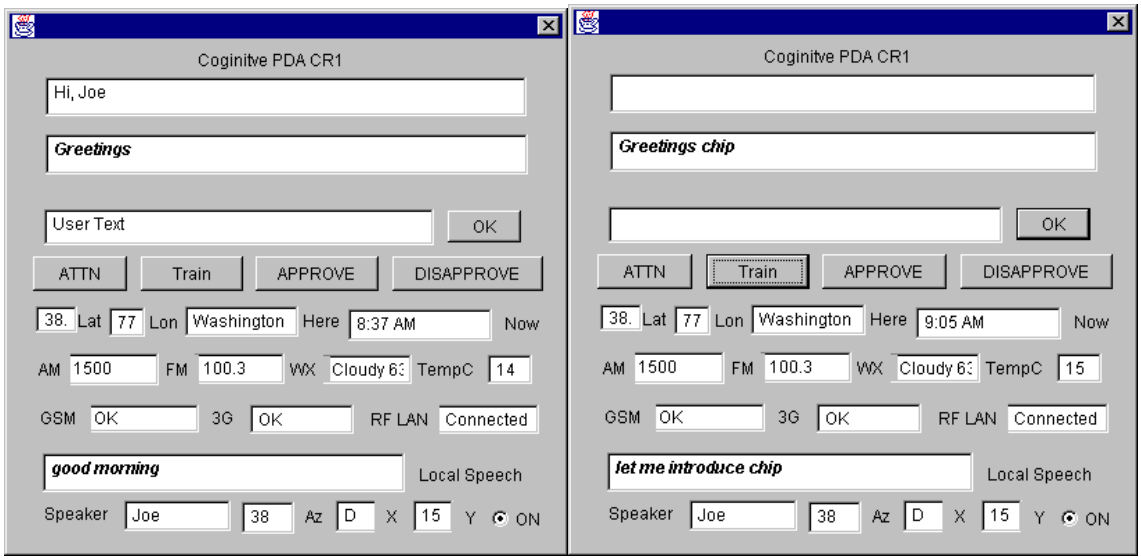
7.1.1 Detecting Basic Communications Contexts

Cognitive radio should reliably identify user communications context so that radio functions can be managed in support of differentiated user needs. Needs and priorities on a trip to the shopping mall are to be differentiated from those in a medical emergency, for example. When shopping, an RF LAN might be preferred to 3G, but in a medical emergency, the urgency of the situation could outweigh the cost of 3G. In addition, wireless service providers benefit from traffic shaping in which large attachments to email are delayed from peak- to off-peak epochs when the user context of the traffic permits. To manage offered traffic in a way that is acceptable to users requires that the PDA learn use-contexts and related communications needs. Context knowledge needed by CR1 was extracted from language learning materials [186], as suggested by Maguire. These materials contain prototypical dialogs in the speaker's native tongue and in the foreign language. The prototypical contexts define user-interactions that may be mapped to generalized communications context. They include those items listed in Table 7-2.

Table 7-2 A Priori Use-Context Relationships

Use Context	Observable Features
Travelling	Departure, arrival, passport control, customs, baggage, changing money, seeking directions, planes, trains, automobiles, taxi, car rental, reservations
Hotel	Check in, registration, staff, telephone, difficulties, laundry, grooming, check out
Eating out	Meal times, cuisine, service, menu, starters, soups, salads, … desert, the bill
Sightseeing	Finding, admission, who/what/when/where
Friends	Introductions, weather, invitation, dating
Relaxing	Cinema, opera, nightclubs, sports, beach
Shopping	Shops, bookstore, clothing, grocery, …
Money	Banking, post office, terms, telegram
Medical	Doctor, body parts, accident, illness …
Other	Maps, basic grammar, general expressions

CR1 has been trained on introductions, eating out, and needing a doctor. Each of these user states has associated information-management tasks for the PDA, many of which include the use of radio resources. For example, if greetings are detected, the PDA attempts to identify names by binding the dialog to prior cases. It may then ask a net-based search engine for the person's e-mail address, home page, on-line business card, etc. CR1's response to a simple greeting is to recognize that a greeting is being offered as in Figure 7-2 (a).



(a) Recognizing a greeting context (b) Responding to an introduction

Figure 7-2 Recognizing and Responding to Greetings

In order to generate the response to Chip, the PDA matches the current input sequence "let me introduce Chip" to all of its prior phrase level knowledge, yielding the following internal data structure ("l.e.t,m.e," is the delimited internal format of the external words "let me"):

Layer3 >Pair(PDA 0 2 7 l.e.t,m.e,i.n.t.r.o.d.u.c.e,?, l.e.t,m.e,i.n.t.r.o.d.u.c.e,f.i.r.s.t.n.a.m.e)9

Since Chip is an unknown name, its partial-sequence correlator at the phrase level inserts a ? in place of the unknown word. It binds this parameter to "**f.i.r.s.t.n.a.m.e**". During training, the

9 The punctuation is inserted by CR1's observation layers, each of which have distinguishing delimiters. The period (.) delimits letters in words, the comma (,) delimits words in phrases, the vertical bar (|) delimits phrases in dialogs, the at-sign (@) delimits context components, and the # sign delimits dialogs in scenes.

use of this phrase is externally reinforced, increasing the relevance of that phrase to this situation. The variable "**f.i.r.s.t.n.a.m.e**" is then bound to "Chip." Most natural language processing systems tag the parts of speech. For example, the Tilburg memory-based speech tagger [187] yields:

let/VB me/PRP introduce/VB Chip/NN ./.,

where: NN Noun, singular or mass; VB Verb, base form; PRP, personal pronoun.

Processing such a tagged structure requires an intensional model of natural language and of the task domain. Cognitive radio architecture accommodates such approaches. CR1's case-based machine learning approach, on the other hand, treats each stimulus as a problem to be solved by retrieving an appropriate stimulus-response case. Each case is called a stimulus-response sequence. Each related collection of such cases is called a stimulus-response model, an srModel. For example, the following simple srPair is from CR1's Hearsay srModel:

Hearsay: l.e.t,m.e,i.n.t.r.o.d.u.c.e,f.i.r.s.t.n.a.m.e ® say, now, here, G.r.e.e.t.i.n.g.s,f.i.r.s.t.n.a.m.e

This variable **f.i.r.s.t.n.a.m.e** is a word-level sequence that is part of a phrase-level stimulus. The phrase-level response is "**G.r.e.e.t.i.n.g.s, f.i.r.s.t.n.a.m.e.**" Such sequences are one of the fundamental data structures of CR1. Applying this case to slightly unfamiliar situations requires several binding steps. First, the binding of "?" to "Chip" occurs in the observation phase. Since Chip is a previously unobserved word-level token, it is bound to "?". Next, there is a partial match between the known phrase **l.e.t,m.e,i.n.t.r.o.d.u.c.e,f.i.r.s.t.n.a.m.e** and the new phrase **l.e.t,m.e,i.n.t.r.o.d.u.c.e,c.h.i.p.** In this case-based retrieval, reinforcement value is earned by "**l.e.t", "m.e,"** and "**i.n.t.r.o.d.u.c.e.**" In addition, **f.i.r.s.t.n.a.m.e** accrues binding cost when it is bound to "Chip." It then takes on the first name role in the response, which is a plan to generate the reply. In this instance, the intended reply "Greetings Chip" is generated as expected.

The Hearsay model is an example of the shallowest reasoning of CR1's internal models. It is synthesized in a PDANode, a Java neural-network like data structure. PDANodes implement the srModel structure of RKRL. Any frame of RKRL may be realized in such an srModel, and any srModel may be translated into an RKRL frame. The rapid prototype does not synthesize RKRL exactly, but its load/store personality capability saves and retrieves the <handle>, <model>, and <body> parts of the frame. Adding the <context> and <resources> parts is

straightforward.

The strategy behind using such simple reasoning capabilities in CR1 is three-fold. First, wireless PDAs put a premium on computational efficiency. Therefore, the representation of procedural knowledge in a way that it can be used with minimum computational burden (e.g. with minimal intensional processing) is particularly appropriate to wireless systems applications. Thus, the shallow intensional reasoning of the srModel is explored in lieu of the deeper intensional models of many conventional natural language processing (NLP) systems. This is not offered as a realistic alternative to conventional NLP, but as an exploration of how a very limited theory of language (in the srModel) supplies useful dialog in a constrained application.

Secondly, however, such models are very easy to train. In CR1, the knowledge of how to respond to such a greeting is not pre-programmed. The user performs the following sequence:

1. Depress the TRAIN button
2. Enter the following in Local Speech[10]: "Let me introduce Joe"
3. Enter the following in the PDA Says area: "Greetings Joe"
4. Click "OK"

This user-friendly sequence of operations follows the strategy that the user should train the PDA by showing it what to do. The relationship between the Speech-Front (local speech) sensor model and the PDA-Says effector model is called "Hearsay." That meta-level knowledge is built into CR1, but the domain knowledge (e.g. of greetings) is not. This push-button training sequence appropriate for users also is equivalent to the following textual training sequence which was used in training CR1:

"train hearsay let me introduce firstname say now here greetings firstname done "

Thus, a programmer or network that knows of the relationship among the PDA's internal models could present the new knowledge in the form of textual training sequences, e.g., using KQML performatives. In addition, a professional trainer could provide training sequences that

10 The simulation of a speech channel via text illustrates what can be done on noiseless data. The approach could be adapted to streams of phonemic hypotheses for speech applications.

clarify the roles expected in applying the cases ("firstname" is clearer than "Joe" as a variable, although they are functionally identical).

The third reason for experimenting with such simple extensional models is to gain insights for the definition of an architecture supportive of the interactive acquisition of intensional models by evolution from simple extensional models. Sequence binding provides an opportunity to acquire an incremental chunk of intensional knowledge. In many case-based reasoning systems, the dimensions of the case are defined a-priori, typically in a database [52]. The raw data is preprocessed to extract the parameters of the case database. This can be a labor-intensive process, or an automated process that is prone to error. The approach taken in CR1 is to integrate this aspect of case generation into an interactive machine-learning framework. In particular, the binding of the stimulus phrase "let me introduce firstname" to "let me introduce Chip" processes this phrase-level sequence as if it were the lambda expression [74]:

l **(x) ("let me introduce " & x)** from the two examples.

The binding process represents substantial information, the surface structure of which is exploited in CR1. The variable firstname could become a discourse variable embedded in an existing srModel, or it could become a new srModel. CR1 makes provision for the machine learning of such variables but does not fully implement the automatic creation of new srModels.

In addition, "Chip" can be associated with the context. Since RKRL axiomatizes space and time, CR1 can be trained to uniquely identify Chip by the time and place of the introduction, differentiating among future Chips. It could also characterize Chip by the times and places at which Chip appears and disappears. This type of association requires more complex reasoning.

7.1.2 More Complex Reasoning: Spatial and Temporal Composition of srModels

Reasoning more complex than Hearsay is constructed by the composition of srModels. Composition may be accomplished temporally or spatially. Spatial composition is the interconnection of multiple srModels so that the responses of one node are the stimuli of another node. Temporal composition is the sequential processing of a response from an srModel as a stimulus to that same srModel. (In order to stay within the bounds of computational stability developed in Appendix B, such feedback is strictly limited to primitive recursion or bounded

minimalization, not arbitrary recursion). Temporal composition occurs when one of the PDA phases (e.g. Plan) iteratively stimulates a specific srModel. At the dialog level, for example, the detection of an introduction could stimulate the completion of a dialog-level sequence:

Dialog1: i.n.t.r.o.d.u.c.t.i.o.n ® r.f.l.a.n, n.o.w, h.e.r.e, t.c.p, q.u.e.r.y, f.i.r.s.t.n.a.m.e| r.f.l.a.n, n.o.w, h.e.r.e, t.c.p,r.e.s.p.o.n.s.e, l.a.s.t.n.a.m.e| s.a.y, n.o.w, h.e.r.e, a.r.e, y.o.u, l.a.s.t.n.a.m.e.?|

The communications goal in this example is to learn the person's last name by querying a VCARD database on the local RF LAN. The sequence Dialog1 consists of multiple phrases. Each phrase consists of a command, a context, and a body. The command is a word or phrase that is recognized by the PDA Phase nodes. For example, the command "**s.a.y"** is included in the srModel of the PDA Phases "orient," "plan," "decide," and "act." In this case, the command is also the name of an effector. The **r.f.l.a.n** effector sets up the protocol parameters (e.g. t.c.p) and posts the query to the simulated RFLAN. In temporal composition, the PDA attempts to "complete the sequence." Completing sequences is the CR1 default plan, executed sequentially as illustrated in Figure 7-3.

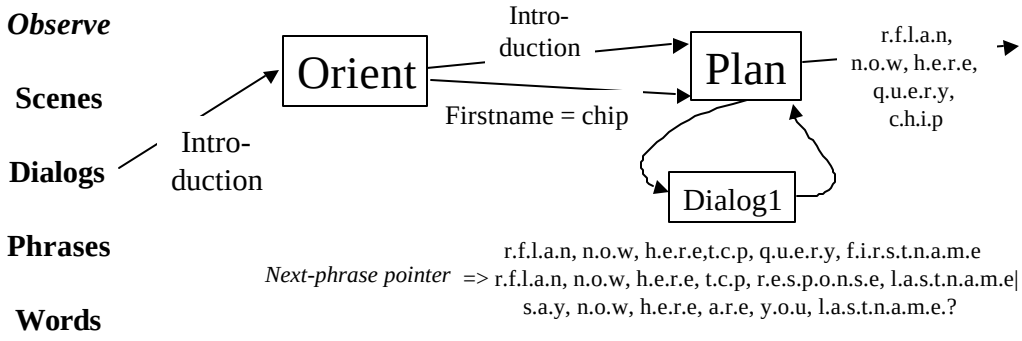


Figure 7-3 A Plan-Phase Node May Iterate Through an srModel Temporally

Dialog1 is composed temporally as follows. The PDA poses the query for firstname to the (simulated) RF LAN. Modeling the interchange with the network as a dialog, it expects to receive a KQML query response that includes "lastname." The response may be the distinguished reply NULL, or it may be a KQML-formatted list of lastnames. The third phrase of Dialog1 causes the PDA to pose the question via its speech synthesizer "are you Chip Maguire? In the rapid-prototype, the emphasis is on the definition of the knowledge representation architecture and interfaces. These examples suffice for that research objective. In a more advanced system, WebL's document calculus [85] could be used to form the query and

consolidate multiple responses. A natural language processing component could then pose a more sophisticated question from the WebL structures. Dialog1 defines the RF LAN interface.

In addition, if the PDA detects local conversation before the RF LAN response is available, it detects a planning conflict. CR1's planning capability is limited. It can perform multiple similar tasks at once, such as saying two different things at the same time. If that fails failing, it can only ask the user for help. The cognitive radio architecture, however, is designed to admit a wide range of alternate strategies. For example, the architecture permits triggered parsing, e.g. to attempt to ascertain the lastname from the speech stream if 'chip' occurs again. The depth to which Chip is understood would be a function of the degree to which CR1's srModels have been exposed to that person. In any event, that person's association with communications context is learned, but other general knowledge is not. Since the emphasis of this research is on wireless data communications, CR1's internal srModels have been structured for and trained on aspects of wireless, such as the selection and parameterization of air interface waveforms.

7.1.3 Selecting Waveforms Using an srModel

CR1 has been trained on radio modes including notional RF-LAN, GSM, GPRS, and 3G parameters. It is trained via the speech/ text interface, e.g.:

srModel 3G: Data Rate → 128 kbps; Cost-per-Mbit → $0.20

srModel GPRS: Data Rate → 32 kbps; Cost-per-Mbit → $0.07

Since GPRS and 3G have srModels from RKRL 0.3, CR1 associates Data Rate as a GPRS model stimulus and 32 kbps as its associated response.

Reasoning about these parameters requires the information flow illustrated in Figure 7-4. Text is provided both to the email system and to the srModel of the email system through the observation phase processing (not shown in the simplified figure). The e-mail model characterizes the email in metrics useful to the messages-model. The messages-model generates the size of the email in Mbits. The email system model defines adjustments to this message length. Compression coding reduces the message length, while FEC expands it.

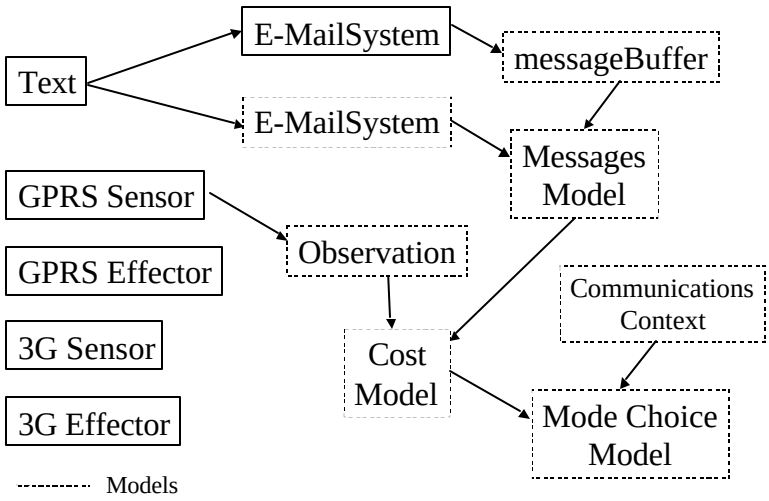


Figure 7-4 Flow of Mode Choice Decision

The GPRS sensor provides its status of that radio link to the Observation phase where it is parsed (through the operation of observation-phase PDA Nodes) and made available to the cost model, which is in the plan phase. The cost model combines the size of the buffer from the messages-model and the data rate from the GPRS sensor status to determine the cost of transmitting on GPRS in its current mode. The mode-choice model, part of the decision phase, then selects among candidate modes to determine which mode is best for transmission. In general, this decision depends on communications context including the PDA's location, predicted duration of the present mode, time to reach a better mode (e.g. RF LAN), etc., according to the criteria provided by the network and/or the user. In CR1, the choice is location-dependent.

The flow of this model illustrates the structure of radio knowledge in CR1. Each sensor, effector, and PDA subsystem (e.g. modem, INFOSEC, etc.) has both a software (and/or encapsulated hardware) module that performs the radio function and a related model that supports reasoning about that function. The related model performs a shadow-function, continually observing the SDR module's inputs and outputs. It thus monitors the behavior of the module and provides structured inputs to the appropriate components of the cognition cycle. The related models are those defined in XML in RKRL 0.3. After experience using these models in a given locale, a CR1-follow on could write those details in XML for web-based knowledge exchange, or it could generate a KQML "tell-one" sequence to a host network.

7.1.4 Setting Waveform Parameters by Concatenating Models

Selection of waveform parameters in the decision phase is also a function of use context. The observe-phase processing integrates the user context (RKRL 0.3's definitions of greetings, shopping, eating out, traffic accident, etc.) with current location and time. In a spectrum rental scenario, the waveform parameter control also would be based on propagation modeling. In conventional cellular radio, the network power control and handoff algorithms manage mutual interference. In the spectrum rental scenario, each mobile unit has to manage power pro-actively to avoid jamming legacy users when joining the rental network. These contextual inputs would drive models that control selected states of the waveforms. The controlled entities include the SDR Forum radio-entities defined in RKRL 0.3, including the modem. These flow through a CIR/dataRate Model that CR1 has acquired by supervised learning. The information flow is illustrated in Figure 7-5.

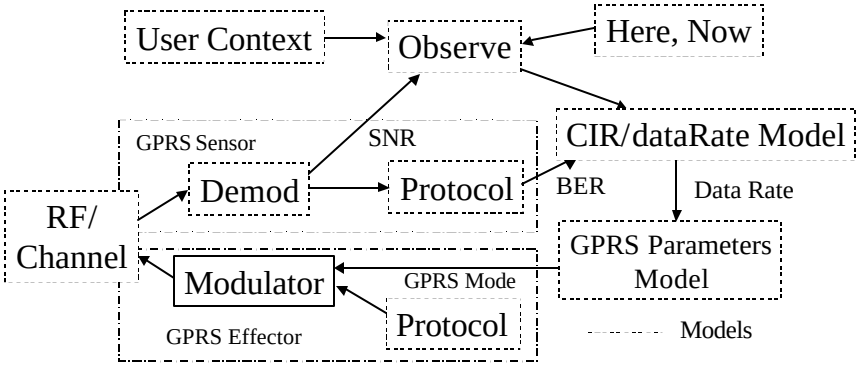


Figure 7-5 Waveform Parameter Nodes Control Action

In this case, the CIR/dataRate Model determines the data rate that can be supported for a given CIR and communications context. The context imposes constraints on radiated power, for example. In addition, radiated power or data rate may be constrained at specific locations or times of the day. In the spectrum rental protocol of Appendix D, for example, the spectrum owner may specify power constraints as a function of transmitter location, activity level, or time of day. Illustrative srModels are as follows:

srModel CIR/dataRate: (CIR, Now, Here, User-context) → @dataRate

(24, 5AM, Autobahn, Any) → 21.4; (18, 6AM, Autobahn, Any) → 21.4

(12, 7AM, Autobahn, Any) →9.05;

srModel GPRS Mode: (@dataRate, User-context) → @GPRS Mode

(9.05, Any) → CS1; (21.4, Emergency) → CS1

(21.4, Any) → CS4

The performance is context-sensitive. If a user emergency is detected, the GPRS Mode Model selects the low data rate mode (CS1) when high data rate is possible, to maximize the probability that the Emergency 911 call will be received error-free. These srModel sequences behave like simple production rules, propositional calculus, or expert-system rules with minimal bindings. From such sequences, one may express a plan as a higher level sequence.

7.1.5 Waveform Parameter Control by Executing a Plan

The waveform parameter control models are equivalent to a planning sequence. That is, concatenated srModels may be (algorithmically) consolidated into an aggregate sequence, and conversely. That sequence may then be executed sequentially, with a corresponding increase in processing time, but using fewer PDANodes. This could occur as follows:

srModel Plan:

Set-Mode @Now@Here@GPRS →

CIR@CIR@Now@Here@Context| GPRS-Mode @dataRate@Context

This plan to set the mode of GPRS consists of two phrases, delimited by "|". The first phrase ascertains the data rate for a given SNR by binding to the following sequence:

(CIR, 24, 5AM, Autobahn, Any) → (@dataRate, 21.4);

This is a template for the srModel CIR/dataRate from the prior discussion. The response from this sequence is a Pair, a Java ObjectSpace object that has two components. The pair is written like a vector. This Pair is the equivalent of a variable binding step. It expresses the binding of the first element to the second element. This links the response of the first sequence implementing the plan to the stimulus of the second phrase. The interpreter instantiates the sequence @dataRate → 21.4 from this Pair. The local variable @dataRate is bound to 21.4 in the srModel Plan by this action. The second phrase, GPRS-Mode @dataRate@Context therefore binds to:

(GPRS-Mode, 21.4, Emergency) → CS1, if an emergency context has been detected.

The result of these three srModel actions is the same as the concatenated operation of the CIR/dataRate model and the GPRS-Mode model. Concatenated models may be implemented in separate processors, creating an inference pipeline with short, deterministic processing delay. The equivalent iterated models store procedural knowledge more compactly, that is, using fewer processors. A plan may be explained in a training session as a long sequence. That training session will be a Scene consisting of a concatenation of dialog sequences. A long sequence of one node is therefore computationally equivalent to concatenation of short sequences. This is similar to human skill acquisition. When initially acquiring a skill, the procedure is explained as a sequence of steps. Subsequently, that knowledge becomes encapsulated and "automatic." Although the automatic transformation between long sequences and concatenated short sequences is not implemented in CR1, its use of these sequences as the primary knowledge representation mechanism creates an architecture compatible with this enhancement.

7.1.6 Shaping Wireless Demand: Planning Primitives

When a PDA downloads a large file such as a detailed map to a restaurant, it may be destroyed or stored for future reference, e.g. in the home computer to conserve space in the PDA. The PDA could send GPRS or 3G email to the user's home to store the map. A user's willingness to pay for an expensive wireless connection may be a function of the content of the map and of the nature of its use. In addition, as the PDA's memory resources are consumed, it must generate a plan to replenish them (e.g. upload the least-used data to the home PC). The PDA may infer that the user will be home in 30 minutes, where the cost of the wireless connection is zero. In such situations, the PDA should apply relevant goals, such as minimizing communications costs. CR1 has an internal planning component that uses a simple planning language to represent plans and to resolve plan conflicts. A plan in CR1 is a structured text sequence:

<Plan> := <Plan-Phrase>| <Plan-Phrase>*

<Plan-Phrase> := <Request> <When> <Where> <What>[11] |

11 This form of plan-phrase does not include user context, while the form on the next line does.

<Request> <When> <Where> <Context><What>

<Request> := Say | Transmit | Receive | Go To | Wait |Dial | Connect

<When> := Now | <Day> | <Month> | <Year> | <Hour> | <Minute> | <Second> | <DateTimeGroup>

<Where> := Here | <Place>|<Lat><Lon>|<X><Y>

<Context> := Any | Normal | Unknown | Introductions | Home (Inside) | Home (Outside) |

Commuting | Travelling | Hotel | Eating out | Sightseeing | Friends | Relaxing |

Shopping | Money | Medical | Other | <New-Context>

A plan may consist of multiple plan phrases. Each request corresponds to an operation that can be accomplished by an effector. Alternatively, a request may correspond to the sensing of an action performed externally. The GoTo and Wait effectors ask the user to go to a place or to wait until a specific time or for a specific duration. <Place> is a known place name, <What> is an arbitrary string, e.g. a message or model body.

The axiomatization of time in RKRL establishes the way CR1 compares times and time delays. Time is associated with every stimulus and response presented to CR1. Thus, the delay between repetitions of characters, words, phrases, dialogs, etc. are associated with each stimulus-response sequence. These may be included in the detection of user context and in machine learning. In the rapid prototype, this facility is under-developed, consisting of the capability to measure all the delays in terms of local clock ticks (not wall clock time).

When CR1's planner detects a conflict in user goals, it proposes an alternative. In CR1's rapid-prototype implementation, it can propose a delay of an action until a specific time or place. For example, when the user attempts to send a large email file while on the autobahn (detected by GPS coordinates), the Cost model introduces a plan to wait until RF-LAN connection is available. This may be at Home or at Work, depending on the time of day (morning commute or afternoon commute). The planning conflict is represented by comparing the plan sequences related to the user input versus the plan sequence from the user's mode choice model:

User Input Generates the internal plan: Send @ Now @ Here @GSM <message buffer>

The ModeChoice model generates: Send @ Now @ Here @RF-LAN <message buffer>

They are reconciled with: Send@Home@5PM@RF-LAN <message-buffer>

This elementary level of planning suffices to shape the traffic on the network as analyzed in the use case scenario.

7.1.7 Mediating a Download Protocol

The last of the use-cases entailed assuring stable downloads of SDR software modules. The SDR Forum has published its download specification [21]. Table 7-3 shows the initial message exchange. The contemporary software-development process might represent in a message sequence chart, using SDL [5]. This human-readable form is then compiled, e.g. into C or C++. In the process, the explicit domain knowledge represented in the SDL and its embedded annotations is lost. Retaining SDL in the operational system presents an alternative that requires substantial processing infrastructure. This is in part because SDL tightly integrates procedural knowledge (e.g. the sequence of steps) with functional knowledge (e.g. the design of operations like "Acknowledge Download Request."

Table 7-3 Illustrative SDR Forum Download Message Sequence (simplified)

NETWORK	Step	TERMINAL
	Initiation	
Page terminal Initiate Download Request		Acknowledge Download Request
	Mutual Authentication	
Authenticate Terminal		
		Authenticate Source
	Capability Exchange	
Request Capability Data		
		Capability Response • Terminal Configuration • API structure supported by terminal • **Hardware Resource** Capabilities • Resident **Software Profile** • Resident software/operator licenses
IF an **appropriate software module** exits, open download channel, ELSE Terminate download		

Hardware Resources include Program Memory, Data Memory, Processing Power, Installed Peripherals, Real-Time capability, etc
Software Profile includes Program/Data memory and processing resources required per resident entity.
An appropriate software module complies with current terminal capability and configuration

A cognitive PDA, on the other hand, partitions the download knowledge into an srModel and a set of effectors that invoke the appropriate functional modules in the appropriate context. The following illustrates how to model this download protocol using CR1:

GSM@Download-request → (Download, Start) | GSM Download-acknowledge()

GSM@Authenticate@Start → (Download, State1) | GSM Authenticate-source()

GSM@Capability@State1 → (Download, State2) | GSM@Terminal-capability |

GSM Send | GSM@API | GSM Send | GSM@Hardware Resources | GSM Send |

GSM@Software Profile | GSM Send | GSM@Licenses | GSM Send

GSM@Terminal-capability → GSMTC32-077-033 (etc. for each capability)

GSM@Software Profile → GSM Baseband | GSM Modem | GSM INFOSEC

Other states terminate the download sequence and perform error checking. A full implementation of the SDR Forum download protocol would require about 50 stimulus-response sequences. Much of the knowledge, such as the software profile, is maintained in the srModel Download where it is available for more flexible KQML interchanges with a cognitive network or peers. In addition, a high-performance natural language processing and database schema system (e.g. patterned after [188]) could be embedded in such a PDA. Sequences like those of Table 7-3 may translated into srModels based on such schema.

The Java software architecture that supports the machine learning, services support, and download knowledge outlined above is now described.

7.2 The CR1 Java Environment

This section describes the CR1 development environment and the software components from which CR1 was constructed. CR1 consists of 40 Java classes. A dozen of these create the simulation environment. CR1 also instantiates about 60 srModels. CR1 is built using a PDA design language specific to CR1. This language could be invoked during operations. Although CR1 does not have that capability, the foundation is laid for future self-extensibility using model-acquisition techniques that call the PDA design language.

7.2.1 CRHome

CRHome is the Java Swing container that controls the CR1 environment. As shown in Figure 7-6, CR1 buttons allow one to set up a simulation, to design, and to run an arbitrary PDA built up using CR1 nodes. It also permits the inspection of the internal structure of the system.

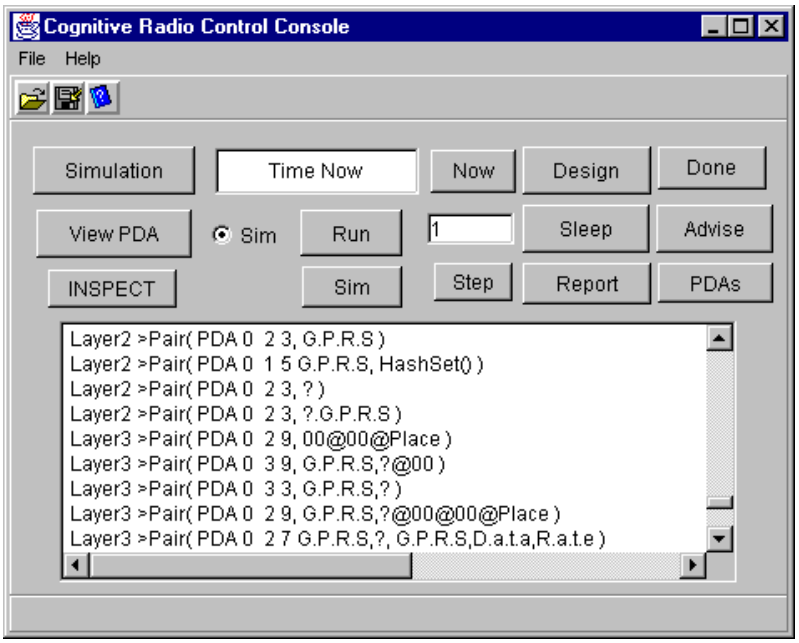


Figure 7-6 CRHome

7.2.2 Simulation

The simulator control designates files to which CR1's environmental stimuli and generated responses may be saved. The data is contained in a static public variable of CRHome called localEnvironment.

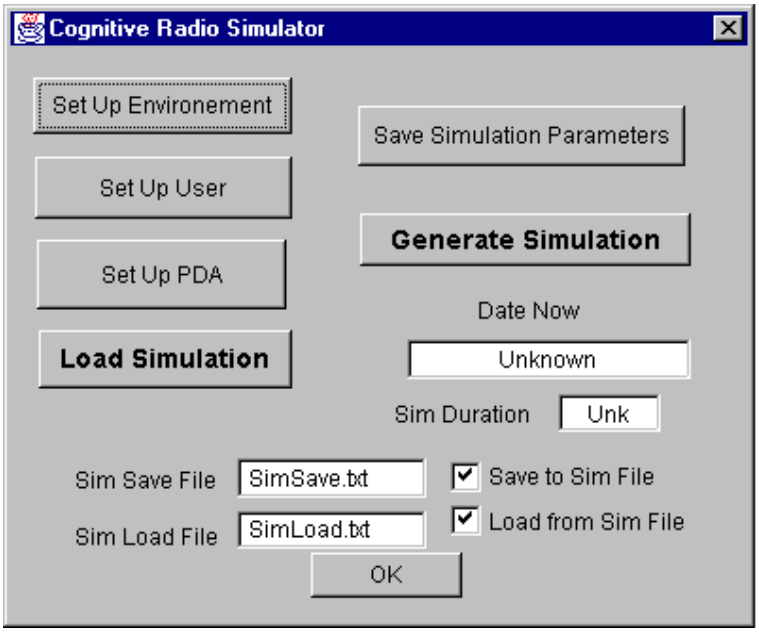


Figure 7-7 Simulation Control Window

The screen that controls parameters that may be set up in the simulator is shown in Figure 7-8. Not all of these parameters are fully functional in CR1, but the available subset supports the use cases.

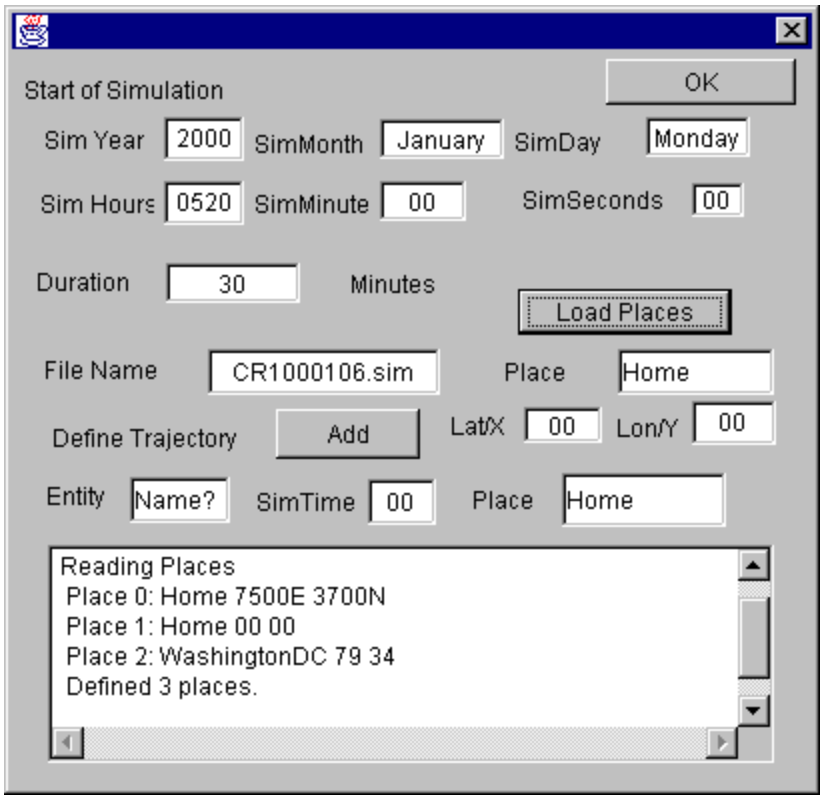


Figure 7-8 Simulation Setup Screen

Items defined in this view become globally accessible via CRHome.

7.2.3 PDA Inspector

The PDA inspector permits the developer to examine the internal states of the PDA. Its scrollable window displays PDA node names, process, node number, and class. The embedded models ("srModel") can be viewed by entering the process number and node number. In addition, the PDA's personality, consisting of all the embedded models, can be stored to a text file and previously saved PDA personalities can be retrieved.

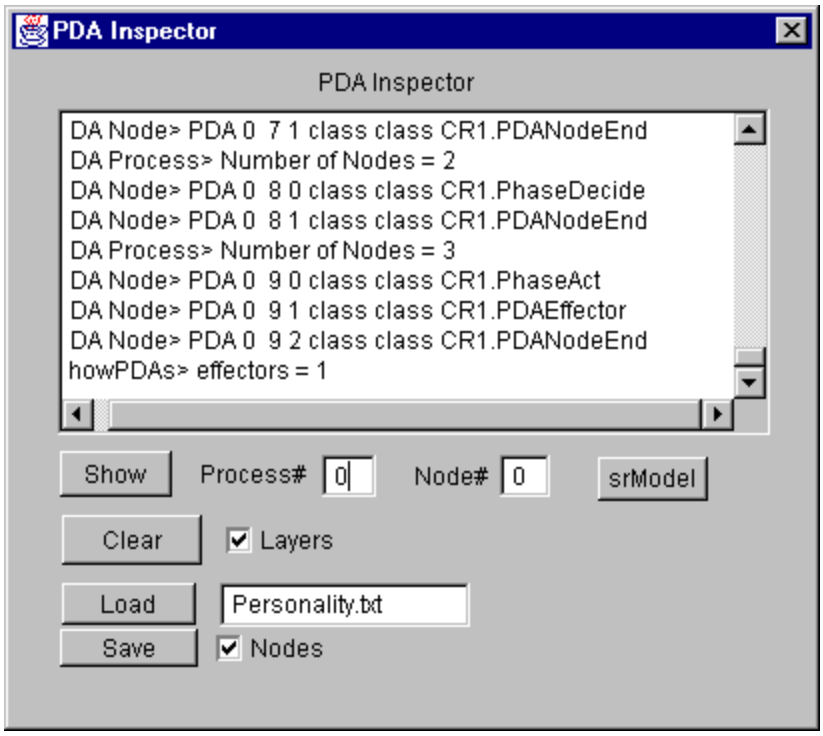


Figure 7-9 PDA Inspector

The inspector facilitates investigations into the role of a-priori personalities in interpreting sensory stimuli. Each srModel is implemented as a finite number of pairs, where a pair is an ObjectSpace data structure: Pair(<stimulus>, <response>), where <> are objects of any sort. CR1 uses strings and other pairs (e.g. binding lists) as srPairs. The response of an srModel to a stimulus is F(srModel . get(<stimulus), srCount.get(stimulus), srDelay.get(stimulus)), where F is the response function defined by the node's runNode algorithm. For more complex nodes, the response is a function of a list of these responses or of the links between the atomic stimuli and known sequences.

7.2.4 PDA Process Organization

The CR1 rapid prototype is implemented as the Java object protoPDA. All PDAs are members of the class PDA that hosts the PDA design components. Processing within the PDA is organized into pdaProcesses that are carried out sequentially for each of the cognitive radio's phases and epochs. The protoPDA acquires a phrase-sized block of stimuli from all sensors before attempting to interpret any of it. The speech, text, and radio channel stimuli consist of at least one phrase, but each channel may offer multiple phrases as well. Block processing reduces

ambiguity and allows set-associative recall to operate efficiently. The organization of processes of protoPDA CR1 is illustrated in Figure 7-10.

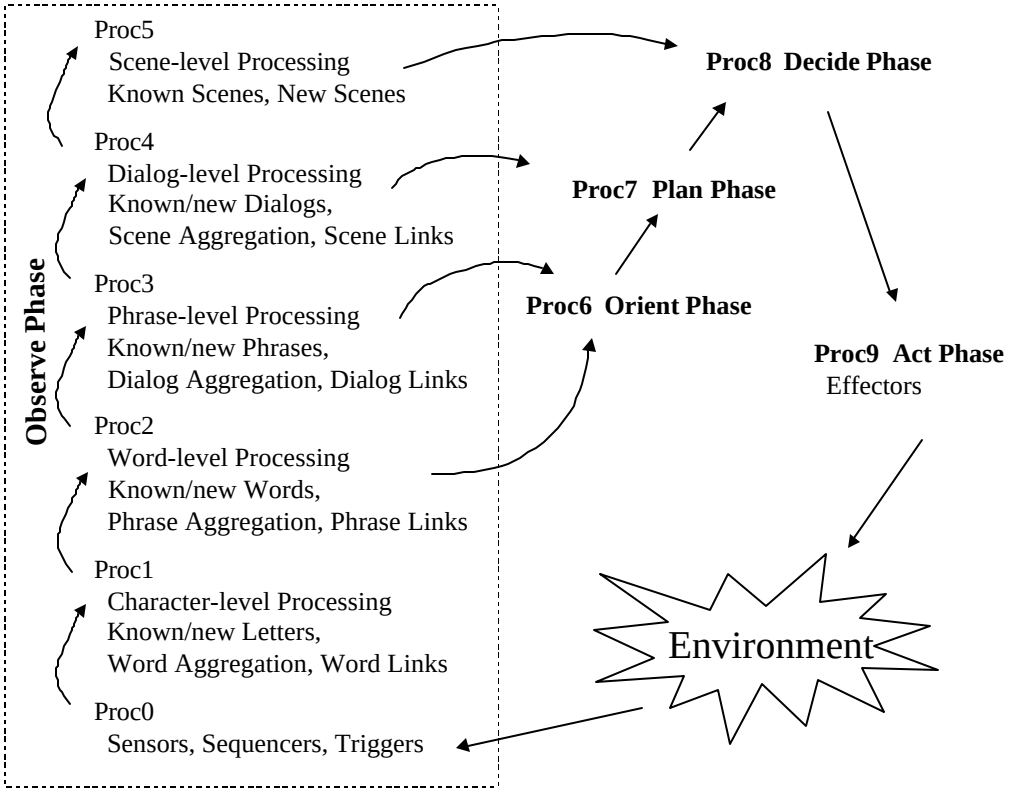


Figure 7-10 CR1 Process Organization

Processes 0 (Proc0) through 5 structure all stimuli into hierarchical sequences of words, phrases, dialogs, and scenes. Processes 0 through 2 are performed for all text, speech, and radio stimuli. In CR1, the radio status sensors provide the interface to the simulated modems and protocol stack. Phrases parsed from these channels are aggregated into phrase-level contexts. Word and phrase "roles" are detected in the srModels Word Role and Phrase Role. These models cue the Orient phase about the communications context (e.g. training, greetings, eating out, etc).

These models are similar to the word-sense attributes of lexical-functional grammars [74]. CR1 does not entail machine-translation level of competence with grammar. Instead, its natural language capabilities are oriented towards inferring communications context and controlling software radios. The natural language processing capability is therefore limited to detecting

contexts from words and phrases and to following explicit instructions. CR1 also learns to recognize the user's stylized patterns of expression, but only in the limited domain of using wireless. In addition to representing human dialog, the word and phrase structures represent protocol sequences such as the high-level download protocol.

The pdaProcess nodes control the timing of internal PDA operations. Processes execute sequentially, running the nodes in a specified sequence. Processes can request that some other process run by placing a request in a global list. Any process with such a request pending will run. The completion of word-level aggregation results in a phrase-level "time-tick" request, for example. The detection of word roles triggers a request for orient-phase processing. Each observation phase process has about 10 nodes, while the other phases have only a few specialized nodes.

7.2.5 PDA Node Organization

PDA Processes Consist of PDA Nodes Organized in Layers. Each pdaProcess is organized to transform concepts from atoms to concept clusters in layers, each of which is organized as in Figure 7-11. Each layer maps lower layer elements onto higher level sequences. This follows a bottom-up flow up the concept hierarchy. Known sequences may be activated in any of these layers. In addition, new sequences are assimilated by the novelty nodes. The processing of stimulus sequences occurs in time windows. In the rapid-prototype, different amounts and types of white space delimit conceptual structures. Long segments of white space delimit multiple layers up the hierarchy, closing sequences. Black space is the inverse of white space. Objects occupy black space, while the (temporal) distance between objects is measured in units of white space.

At the bottom level of the layer, elements are detected by the layer's srModel of Known Elements. Since these elements may correlate partially to known sequences, this model's response stimulates the partial sequence detector, implemented in the Java construct PDANodeLinks. Positive correlation with known sequences yields a response from this node. These nodes bind the partially recognized series of elements to known sequences embedded in the linksModel of the node. The binding is available is expressed as a pair (Stimulus-sequence, Best-match) in which the best-match includes bindings of the form @New-element = Known-

element@. The binding first generates the intersection of all known sequences with the current stimuli. If the intersection is empty, then the best intersection is retained. This is the smallest non-null intersection with the elements in the stream. If this set contains more than one element, all bindings are computed, and the best-binding is provided as the response from this node. The best-binding accumulates value for identical matches and cost for binding unequal elements.

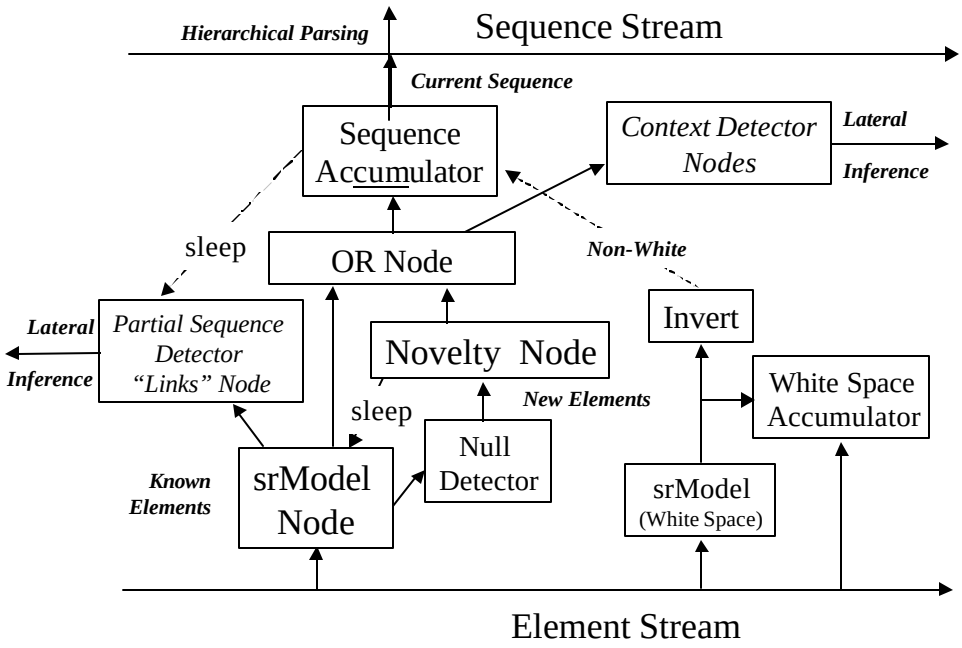


Figure 7-11 Structure of Observe-Phase Layers (simplified)

In addition, a limited-scope dynamic time warping [189] is performed. This permits more than one new element to bind with one known element and conversely. The response from this node contains the bindings necessary for support phase-to-phase lateral inference. Typically information flows laterally from the observation phase processes to the orient-phase process(es).

Novelty nodes in each layer acquire stimuli not recognized by the layer's Known Elements srModel. Any non-null response from this model is accumulated into a higher level sequence. Null responses cause the novelty node to acquire the element and create a response that is the value of the stimulus (identity function). This response is then selected for sequence-level accumulation by the OR Node. Sequence accumulation is implemented in the Java PDANodeCum class. In addition, the stream of known and new elements stimulates nodes that detect context. These are typically srModel nodes with named models. The naming of models permits the automatic acquisition of new stimulus-response sequences into these models. In the

simplest training sequences, the models to be trained are named explicitly by the trainer. In more natural training sequences, the names of the models are inferred by the communications context, implementing supervised knowledge transfer from the user.

7.2.6 Memory-Based Concept Structures

The pattern of accumulating elements into sequences illustrated in Figure 7-11 is repeated at the levels corresponding to words, phrases, dialogs, and scenes. The data structures generated by PDA Nodes create the concept hierarchy of Figure 7-12. These are the reinforced hierarchical sequences. Unless digested (e.g. by a sleep process), the observation phase hierarchy accumulates all the sensor data, parsed and distributed among PDA Nodes for fast parallel retrieval. Since the hierarchy employs memory-base learning techniques [190], this is a memory-intensive approach. In addition, recent research shows the negative effects of discarding cases in word pronunciation [191]. In word pronunciation, no example can be discarded even if "disruptive" to an intensional model. Each exception has to be followed.

Sequence	*Level of Abstraction*
Context Cluster	***Scenes*** in a play, Session
Sequence Clusters	***Dialogs***, Paragraphs, Protocol
Basic Sequences	***Phrases***, video clip, message
Primitive Sequences	***Words***, token, image
Atomic Symbols	***Raw Data,*** Phoneme, pixel
Atomic Stimuli	External Phenomena

Figure 7-12 Concept Hierarchy of CR1

Similarly, in CR1, each stimulus from the user or network is a case in which knowledge may be present. As a minimum, if the identical context occurs again, the PDA will follow exactly the prior course of events. To the degree that the prior situation yielded training, it will respond more appropriately, however. Each occurrence of a stimulus is therefore retained and counted, reinforcing that sequence. The time delay between successive occurrences is also measured. If time delays match, the sequence is further reinforced. Counting successful

stimulus-response pairs occurring throughout the architecture reinforces experience. Counting time delay enables reasoning over time. Sequence elements vary from the atomic to the complex, as a function of the level of the concept hierarchy.

The principle of operation of the concept hierarchy follows from the operation of the PDA Nodes. Atomic stimuli (e.g. from a microphone) are converted to atomic symbols (e.g. phonemes) by preprocessors such as speech-to-text phoneme recognizers. Temperature, pressure, time, location, and radio-channel states determined by lower-level SDR software are processed in parallel. The PDA Nodes process the resulting logically parallel streams of atomic symbols. The observation phase builds a parsed internal representation of the sensor data. Sequences are initially encoded in novelty nodes. After a sleep cycle, they are transferred to the srModels for known elements and sequences, with activation links to constituent elements (e.g. in the Links Nodes). Consequently, the all the fine structure of a time-space experience is encoded in the sequence hierarchy and available for rapid context-sensitive recall.

Sequences are delimited by designated delimiters, typically the most common element(s). For example, white space is the most common character in text, so it delimits atomic symbols (characters) into primitive sequences (quasi-words). The inter-utterance pause is the most common temporal epoch in speech, so this white space delimits primitive sequences into basic sequences, quasi sentences. Since the white-space detector is an srModel, the white-space model may learn that other things constitute white space. Noise background in a speech stream, for example, could be identified as white-space so that the PDA only responds to speech epochs with high SNR. This processing flow has structural similarities to neural networks, but has the advantage of transparent transduction of information, up the hierarchy. Neural networks encapsulate the information in the synaptic weights, reducing the explicit flow of information through layers of neural networks. This approach is an intentional departure into somewhat uncharted territory, driven by the lack of structure of the task domain.

Context Clusters correspond loosely to RKRL micro-worlds. These include communications-context sequences, radio-context sequences, network- context sequences, and/or user- context sequences. This approach yields a proliferation of context clusters. Some may be deleted during the sleep cycle, depending on the type of machine learning incorporated in that epoch. The analysis of memory requirements of context clusters below shows that machine

learning in the sleep epoch should drive the higher levels of the concept hierarchy towards a small number of prototypical scenes (e.g. 100). Each scene may be supported by a larger number of dialogs (e.g. 500) and a reasonable vocabulary (e.g. 2000 words) without exceeding memory capacities available in PDAs within about five years.

7.2.7 PDA Nodes

Each component of the PDA is a member of the Java class PDANode. The class hierarchy is illustrated in Figure 7-13. All of the subordinate nodes inherit the core stimulus-response model (an ObjectSpace HashMap) from PDA Node. These classes each override that node's runNode() method to provide unique behavior. In addition, each node class has some unique internal data structures. Slot-type data structures link nodes to each other. All nodes have a responseSlot which contains its output, as with neural networks. The stimulusSlots may be strings or arrays (e.g. stimSlot).

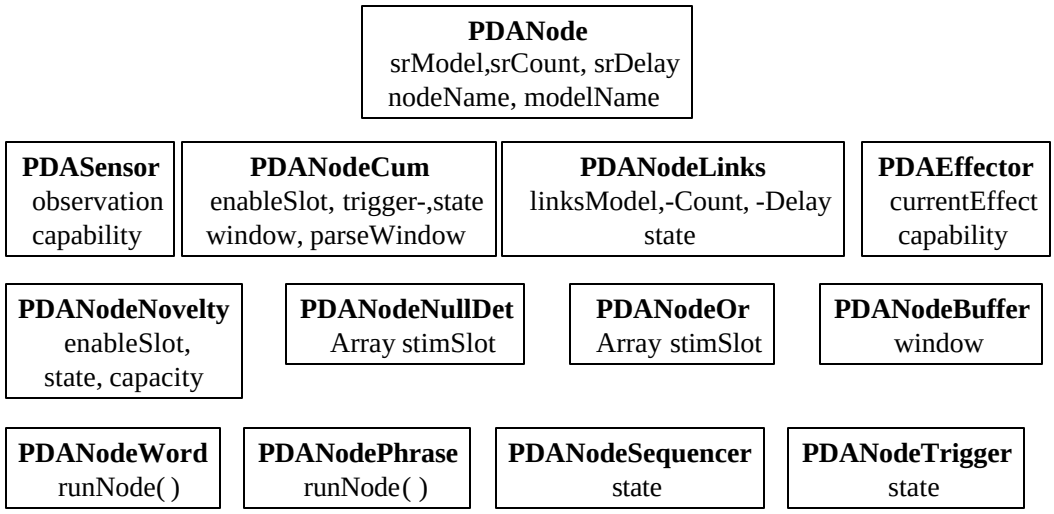


Figure 7-13 PDANode Classes

Sensors extract data from the localEnvironment which has been placed there by either the protoPDA or by the simulation or both. Effector nodes place their effects into localEnvironment from which relevant data is extracted by either protoPDA (in the rapid prototype) or by the simulation (interfaces are provided, but live interactive simulations are not included in the rapid prototype).

When run, the PDANode class checks for a valid input. The flag “INACTIVE” tells the node that it has not been stimulated, so it usually does nothing. Otherwise, it looks up the stimulus in its srModel. PDANodes may also be called srNodes. The stimulus can be any object, as can the response, so this is a very general functional map associative memory. An arbitrary function may be run on the retrieved response before posting it to the response slot. Pre-defined srModels include known words, phrases, dialogs, and scenes, word roles, phrase roles, user, network, and radio models discussed above. All of these models are savable and retrievable using the inspector. Novelty nodes are the companions to srNodes. When an srNode lacks a model, its companion novelty node creates a model of the stimulus, creating the default response that is exactly equal to the stimulus. When a novelty node acquires new knowledge, it places itself on the needsSleep list and decrements its capacity. When capacity meets variable thresholds, the node complains to the Orient phase that it is tired, which advice would normally be passed on to the user. During the sleep cycle, novelty nodes transfer what they have learned into the companion (“partner”) srNode. Sleep does not take a long time (a few seconds for CR1), but the PDA cannot do anything at all while it is sleeping because outside stimuli would interfere with the knowledge consolidation process.

Sequence accumulation is accomplished in the Cum (pronounced quume) nodes. These nodes create srModels in which the elements are the items from its stimulus slot and the sequence is the accumulated value of its parseWindow. After the first stimulus, the Cum nodes insert programmable delimiters into the sequence forming in the parseWindow.

Links nodes are continuously computing the partial correlation of the current elementary stimulus with the known sequences of those stimuli (characters->words, words-> phrases, etc.). When links nodes observe “INACTIVE”, they embed “?” into their accumulation window. If the links node is attached to KnownWords, for example, it inserts ? wherever an unknown word appears in a phrase. Links nodes are given linksModels from the srModel of a companion Cum node. The current prototype computes only the exact (time synchronous) cross-correlation of the stimulus sequence with the known sequences. Links nodes use dynamic time warping to recall sequences that are not well time-aligned, but the time warp examines only a limited number of hypotheses in order to avoid combinatorial explosion in this operation.

Sequencer nodes perform parallel to serial conversion so that each element of a stimulus

block can be processed by the lower level processes. OR nodes perform parallel to serial conversion, aggregating items processed along parallel lower level afferent paths. OR nodes aggregate contexts, for example, by forming speech, text, time, place, and RF-related stimuli into sequences that can be analyzed by the orient and plan phases. The phase nodes transform these aggregated sequences into planning and action sequences. The effectors perform the tasks specified in the action sequences.

7.2.8 The Cognitive PDA Design Language

The prototype PDA is designed from the PDANode components using the design language summarized in this section. The language consists of calls to Java operators that instantiate PDA Nodes from the classes summarized above. Some language statements populate the nodes with capabilities. Although arbitrary a-priori knowledge may be preprogrammed using the design language, this approach is deprecated. In CR1, only the absolute minimum a-priori knowledge was provided, less than two dozen srPairs needed to permit the interactive bootstrapping of all subsequent knowledge via machine learning. This section is essentially an annotated listing of the Java class "Design." This class operates on a PDA List in CRHome. It generates one or more PDA's. The presentation style of this section is to show the Java code in bold and to comment on it. The following is the first code from the Design class of CR1:

```
PDA protoPDA = new PDA();  // PDA shell

protoPDA.putName(protoPDA.makeName());

//PDA 0
```

ProtoPDA is given the name PDA 0. Multiple PDAs would be possible with minor revisions to the Java code. The following sequence of Java creates CR1's sensors, giving them a default value and the capability to access specific data from the localEnvironment. For example, the latSensor has a default value of "00" and the capability to access the information on "Lat" in the local Environment.

```
//Initialize sensors
int latSensor = protoPDA.newSensor("00", "Lat");
int lonSensor = protoPDA.newSensor ("00", "Lon");
int speechFront = protoPDA.newSensor("NO_OBSERVATION", "Speech-Front");
```

This sensor also wants to be processed one character at a time, so it requests proc0 and proc1 by asking for “Character” level time-ticks. The designer does this by setting a “tick-request,” TickRQ, onto the speechFront sensor.

```
protoPDA.setTickRQ("SENSOR", speechFront, "Characters");
```

The other sensors defined for CR1 are as follows:

```
int hereSensor = protoPDA.newSensor ("00", "Here");
int xSensor = protoPDA.newSensor ("00", "X");
int ySensor = protoPDA.newSensor ("00", "Y");
int azSensor = protoPDA.newSensor ("00", "Az");
int rflanSensor = protoPDA.newSensor ("00", "RFLAN");
int pcsSensor = protoPDA.newSensor ("00", "PCS");
int gsmSensor = protoPDA.newSensor ("00", "GSM");
int tempSensor = protoPDA.newSensor ("00", "Temperature");
int wxSensor = protoPDA.newSensor ("00", "WX");
int amSensor = protoPDA.newSensor ("00", "AM");
int fmSensor = protoPDA.newSensor ("00", "FM");
  int yearSensor = protoPDA.newSensor ("00", "Year");
int monthSensor = protoPDA.newSensor ("00", "Month");
int daySensor = protoPDA.newSensor ("00", "Day");
int hoursSensor = protoPDA.newSensor ("00", "Hours");
int minutesSensor = protoPDA.newSensor ("00", "Minutes");
int secondsSensor = protoPDA.newSensor ("00", "Seconds");
int speakerID = protoPDA.newSensor("joe","SpeakerID");
```

The first process created is PDA-0-0, processs 0 on PDA 0. This process handles time-ticks called “characters.” This is accomplished in Java as follows:

```
//PDA-0-0

int proc0=protoPDA.newProcess();

protoPDA.addTick(proc0,"Characters");
```

Nodes are then added to realize the character-level parsing functions. The first node is PDA-0-0-0. It is a sequencer, a node that presents a block of sensory stimuli one character at a

time to subsequent nodes. Although somewhat counter-intuitive, sequential parsing followed by parallel context aggregation yields a consistent set of phrase-level data for the rest of the cognition cycle.

```
//PDA-0-0-0
  int node00 = protoPDA.newNodeSequencer(proc0);
  protoPDA.setModelName(proc0, node00, "SEQ");
  protoPDA.enableTrigger(proc0,node00);
  //Connect the sequencer (node 0,0) stimulus to the sensor at sensorlist(2)
  //setStimulus PDA-0-0-2 <- PDA 0, Sensor 2
  protoPDA.setSensor(proc0, node00, 2);
```

The next PDANode is the srModel for known letters. This needs a name, a stimulus, and a layer in the inference hierarchy on which to place its results. In addition to the layered response, (a pointer to) this node may be placed on the stimulus slot of any other node to make its responses available to that node.

```
  //PDA-0-0-1
  int node01 = protoPDA.newNode(proc0);
  protoPDA.setModelName(proc0,node01,"KnownLetters");
  protoPDA.setStim(proc0,node01,proc0,node00);
  //Give node 01 the ability to report on level 1
  protoPDA.setLayer(proc0,node01, new Integer(1));
```

The white-space detector node is an srModel that has been pre-programmed with the ability to detect delimiters such as blanks. Its stimulus is set to PDA-0-0-0, the sequencer. This establishes the flow of information illustrated in the layer processing above.

```
//PDA-0-0-2 //The white space detector to enable white accumulation
  int node02= protoPDA.newNode(proc0);
  protoPDA.addSR(proc0,node02," ", "ENABLE");
  //Connect sequencer at 0 to drive sr at 0 2, the whitespace node
  //setStimulus PDA 0 0 2 <- PDA 0 0 0
  protoPDA.setStim(proc0,node02,proc0,node00);
  //Timing control node: processes speech looking for x|3 blanks
```

In addition, a series of nodes detect blocks of white space, triggering higher layer nodes to provide a response at the time the white space that delimits these entities is detected. In a real-time system processing speech, these detectors would look for epochs such as inter-syllabic gaps and inter-utterance gaps. Alternatively, a character stream from an existing speech recognizer could indicate the nature and duration of noise epochs. The word trigger node is defined as

follows:

```
//PDA-0-0-3   triggers words
int trig1 = protoPDA.newNodeTrigger(proc0);
protoPDA.setSensor(proc0, trig1, speechFront);
protoPDA.setSpan(proc0, trig1, 2);
protoPDA.enableTrigger(proc0,trig1); //sets state to enable
protoPDA.endTrigger(proc0, trig1); //forces trigger when no terminal blanks
//Timing control node: processes speech looking for x|5 blanks
//PDA 0 0 4      triggers phrases
```

This pattern continues, defining phrase and other triggers. Sensor processing nodes are then defined as instances of word and phrase processing nodes. The word and phrase level nodes parse formatted streams, e.g. from waveform sensors such as the GSM waveform. These streams are not expected to have new, unrecognized items, so the machine learning of the observation layer is deferred in preference for less memory-intensive conventional parser. If the internal channel were considered to be noisy or if the waveform is new (and therefore could present unfamiliar sequences), these less capable nodes could be exchanged for about 10 nodes each that accomplish the parsing function, but with the novelty nodes and accumulators needed for machine learning in the sleep cycle.

```
//PDA 0 0 7  - start of sensor processing nodes
int latNode = protoPDA.newNodeWord(proc0);
protoPDA.setSensor(proc0, latNode, latSensor);
//protoPDA.setTickRQ(proc0,latNode,"Action");
…
```

The next series of nodes define the word-level process, PDA-0-1. This process includes null-detection, accumulator, OR, inverter, links, etc. as needed to complete the first processing layer.

```
//PDA 0 1 // the second pda process
int proc1 = protoPDA.newProcess();
protoPDA.addTick(proc1,"Characters");
//Add a PDANodeNullDet, a node that detects a failure to respond
//PDA 0 1 0
int node10 = protoPDA.newNodeNullDet(proc1);
//Give this node the ability to monitor the first node bank
//addStimulus PDA 0 1 0 <- PDA 0 0 1 //this monitors all the a priori known stuff
protoPDA.addStim(proc1,node10, proc0,node01);
```

Next, a novelty node is defined and linked to the sequencer.

```
//PDA 0 1 1  // PDA Novelty
int novelty1= protoPDA.newNodeNovelty(proc1);
//Install the sequencer node on the novelty node's stimulus slot
//setStimulus PDA 0 1 1 <- PDA 0 0 0
protoPDA.setStim(proc1,novelty1, proc0, node00);
//Install the Null detector node on the novelty node's enable slot
//setEnable PDA 0 1 1 <- PDA 0 1 0
protoPDA.setEnableNvND(proc1,novelty1,proc1,node10);
//Install a partner node for ML in the novelty node's partner
//setPartner PDA 0 1 1 <- PDA 0 0 1
protoPDA.setPartner(proc1, novelty1, proc0,node01);
protoPDA.setLayer(proc1,novelty1, new Integer(0));
```

The partner node to a novelty node is the srModel to which the novelty node will transfer its knowledge during a sleep cycle. The node operates only if its enable slot is active. Otherwise, the slot's value is the distinguished internal string "INACTIVE". This serves as a gate to assure that the novelty node assimilates a stimulus only when the Known Element node does not recognize the element. The next node accumulates the white space elements into sequences of white space. In principle, this allows a protoPDA to process both background and foreground simultaneously. In CR1, only the foreground sequences are processed.

```
//addNodeCumulator PDA 0 1 2
int cum1 = protoPDA.newNodeCum(proc1);
//protoPDA.setTickRQ(proc1,cum1,"Words");
//Install the sequencer node on the cumulator node's stimulus slot
//setStimulus PDA 0 1 2 <- PDA 0 0 0
protoPDA.setStim(proc1, cum1, proc0,node00);
//Install the sr node 002 on the cum node's enable slot this is whitespace
//setEnable PDA 0 1 2 <- PDA 0 0 2
protoPDA.setEnableCum(proc1,cum1, proc0, node02);
//protoPDA.setTickRQ(proc1, cum1, "CUM1");
protoPDA.setLayer(proc1,cum1, new Integer(0));
```

```
protoPDA.setMax(proc1, cum1, 2); //Sets to trigger on inter-phrase? Pauses
```

The details of the inverter and other accumulator node have been deleted. The next step is to initialize the links node. This node was initially populated with some test data that has been commented-out. The word “one” was recognized through its association with the characters “o,” “n.” and “e.” In addition, this node is given a right-delimiter of “.”. This results in and internal format for words that have the embedded periods. This format makes it easy to parse strings at different levels of the hierarchy using a String Tokenizer, a standard Java construct.

```
//PDA 0 1 5
int links1 = protoPDA.newNodeLinks(proc1);
//protoPDA.addLinks(proc1, links1, "o", "one");
//protoPDA.addLinks(proc1, links1, "n", "one");
//protoPDA.addLinks(proc1, links1, "e", "one");
protoPDA.setStim(proc1, links1, proc0, node01);
protoPDA.setLayer(proc1, links1, new Integer(2));
protoPDA.setModelName(proc1,links1,"LetterLinks");
protoPDA.setEnableLx(proc1, links1, proc1, inverter1);
protoPDA.setDelimiters(proc1,links1,"", ".");
//Link this links node to cum2
protoPDA.setPartnerCX(proc1,cum2,proc1,links1);
```

The following sequence of comments shows the nodes of the second layer in the observation phase.

```
//PDA 0 2 // the second pda process
//PDA 0 2 0
int words = protoPDA.newNode(proc2);
protoPDA.setModelName(proc2, words, "KnownWords");
//PDA 0 2 1 newNodeNullDet(proc2);
//PDA 0 2 2protoPDA.newNodeNovelty(proc2);
//PDA 0 2 3  int or1 = protoPDA.newNodeOR(proc2);
//PDA 0 2 4  whitespace enabler
/PDA 0 2 5  int cum3 = protoPDA.newNodeCum(proc2);
```

```
//PDA 0 2 6   int inverter2 = protoPDA.newNode(proc2);
//PDA 0 2 7  - phrase level parsing    int links2 = protoPDA.newNodeLinks(proc2);
//PDA 0 2 8    int cum4 = protoPDA.newNodeCum(proc2);
//PDA 0 2 9 - integrates phrase-level contexts
   int orP = protoPDA.newNodeOR(proc2); ….
//PDA 0 2 10    int wordRoles = protoPDA.newNode(proc2);
//PDA 0 2 11    int end2 = protoPDA.newNodeEnd(proc2);
```

The third, fourth, and fifth layers are analogous to layer 2. Layer 3 processes phrases, layer 4 processes dialogs, and layer 5 processes scenes. Next, each subsequent phase (orient, plan, decide and act) are set in dedicated processes. Process 6 comprises the orient phase, with its primary inference-control node, the Java class PhaseOrient.

```
proc6 = protoPDA.newProcess();
//PDA 0 6 0
   int orientPhase = protoPDA.newPhaseOrient(proc6);
   protoPDA.setWords(proc6,orientPhase,proc2,or1);
   protoPDA.setWordRole(proc6,orientPhase,proc2,wordRoles);
   protoPDA.setPhrases(proc6,orientPhase,proc3,or3);
   protoPDA.setPhraseRole(proc6,orientPhase,proc3,phraseRoles);
   protoPDA.setLayer(proc6,orientPhase,"ORIENT");
   protoPDA.setTickRQ(proc6,orientPhase,"Plan");
   //PDA 0 6 z - end phase
   int end6 = protoPDA.newNodeEnd(proc6);
   protoPDA.setTickRQ(proc6,end6,"Orient");
```

The End node terminates processing for this phase by removing “Orient” from the tick-list. The planning process (proc7), decision process (proc6) and action process (proc9) are defined analogously to the orient phase processes.

7.3 The Wake Cycle

CR1 uses the minimal knowledge of syntax provided in the Design phase to bootstrap knowledge as reinforced hierarchical sequences. CR1 supports both passive and active machine learning during its wake cycle. If the Train button is not depressed, then the phrase in the local

speech buffer ("let me introduce firstname") is parsed and correlated to known words and phrases. Before CR1 has acquired content into its internal models, it passively parses and acquires everything into its novelty nodes, but can not respond. As internal models are populated, it develops the capability to respond as indicated in the use scenarios. This section traces that development of knowledge.

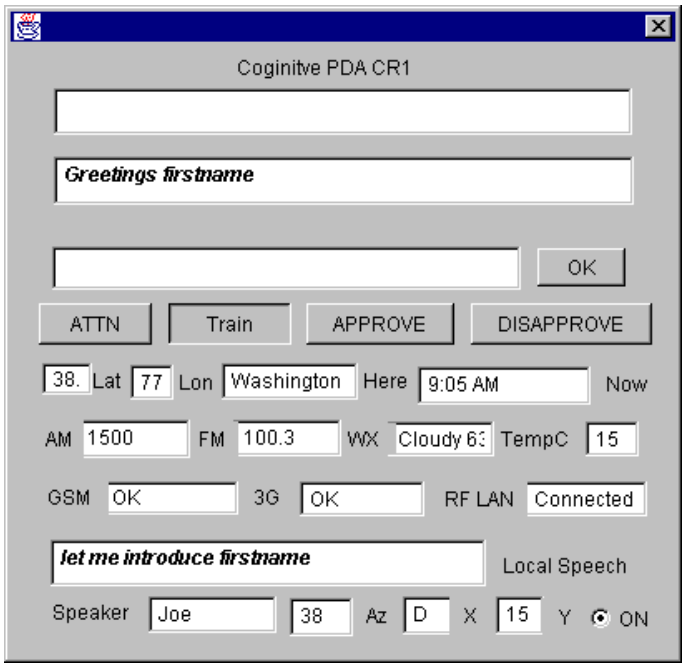


Figure 7-14 Passive and Active Machine Learning

The user can train CR1 the same way that the designer trains it as illustrated in Figure 7-14. The trainer expresses a phrase or sentence in speech or text, edits the PDA's output area, presses the Train button and then presses OK, which serves as the start button for CR1's internal processing cycle. The PDA takes the sensory input as a stimulus to be paired with the response given by the values that the trainer has placed in the effector area. In the example, the PDA is supposed to Say "Greetings firstname" when the expression "let me introduce firstname" appears as speech input. The Train button causes the PDA to ignore all context cues except the ones that the user has edited since the last interaction cycle. This input is taken as context-free. If the latitude and longitude had been edited, it would have learned to associate this response loosely with the vicinity of Washington, DC (which could be helpful in social situations). The first occurrence of such phrases will be reinforced whenever their use yields appropriate positive

feedback. The trainer could have said “let me introduce charlie” but then the phrasodic template for introductions would have used charlie as the variable representing the first name. It is easier to debug the knowledge structures if the items being used as templates look like templates, although that is not at all necessary.

7.3.1 Learning to Detect Contexts

The Berlitz Phrase Book and Dictionary for Swedish was used to train CR1 in greetings and eating out. The srModels thus generated are stored in personality files. Each personality contains the stimulus-response sequences (srSequences) for all the nodes with named models in the PDA. For example, the Introductions personality consists of 56 kB of text consisting of 569 srSequences. These load in less than a second and run handily in real-time interactions with CR1. The introductions training sequence is as follows:

How do you do; Pleased to meet you; may i introduce firstname;
this is firstname lastname; excuse me; john this is mary; john this is joe
may i present mary; my name is charles; my name is rebekka
my name is lars; my name is firstname lastname; what is your name
pleased to meet you; how are you; fine thanks and you; how long have you been here
i have been here a week

These 18 phrases consist of 74 words (36 different words). The 569 relationships represent the parsing of these phrases into characters and words, and the fact that each character is fully indexed (in a Links node) at the word level, each word is indexed into each different phrase, etc.

The process of training the PDA that these phrases are introductions is informative. In rote training of the PDA, the trainer tells the PDA it is being trained, the trainer provides a stimulus-response sequence for an internal model, and says, “done.”

train phrasesense let me introduce firstname introductions done.

An easily parsed natural language version of this sentence is

“The phrase sense of ‘Let me introduce firstname’ is ‘introductions.’”

Somewhat more convoluted, but more natural:

“During introductions, I could say ‘Let me introduce firstname.’”

The subjunctive case "could" signifies that this is a training sequence. This is implemented by training the Word Sense srModel with the case "could → train." The phrase 'During introductions' signifies the response that a phrase sense model should produce. Finally, the fact that 'introductions' is a user communications context establishes that this chunk of knowledge needs to go in the phrase sense model that detects such contexts at the phrase level. CR1 handles the first format of such expressions as illustrated in Figure 7-15.

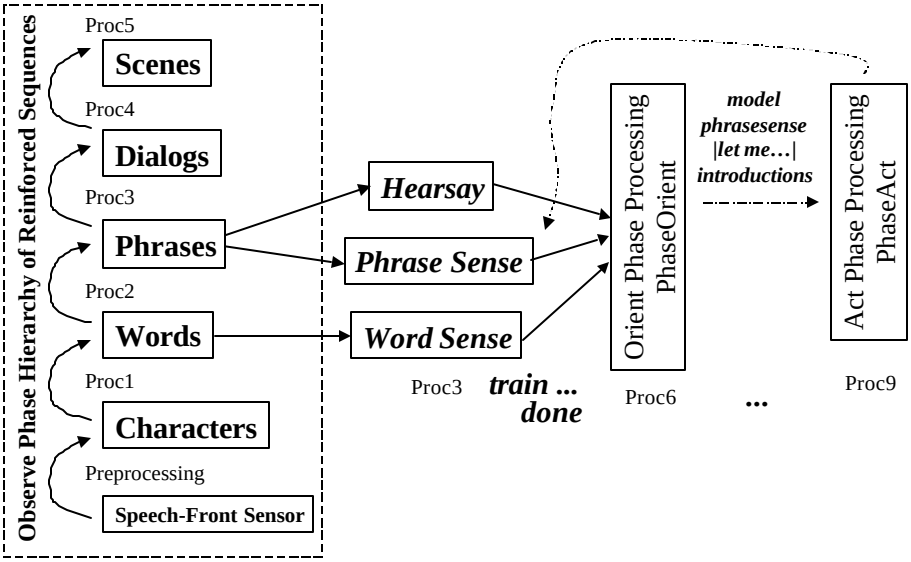


Figure 7-15 Lateral PDA Nodes Detect Communications Context

The models Hearsay, Phrase Sense, and Word Sense are the lateral models that respond to stimuli from the observation hierarchy. The instruction constitutes a dialog-level sequence. The words, "train" and "done," are registered in the Orient phase srModel as warranting a change of internal state during the processing of this dialog. When the PDA detects "train" in this stylized context, it expects intervening phrases until the training delimiter "done" or any of its equivalent expressions occurs. In addition, "introductions" is already known as the name of a model (by a similar prior training sequence). The command "model phrase sense | let me introduce firstname | introductions" is synthesized by the orient phase. The context @now @here is also generated. The context is resolved and stripped from the command by the intervening plan and decide phases. The act-phase passes this command to its PDANode Actuator that has been given the capability to modify any named internal models. The new phrase sense is therefore added to the phrase sense model. Subsequently, if that phrase is used, it evokes "introductions" into the orient phase.

Other models in CR1 are trained using this machinery. The result is that CR1 bootstraps its knowledge from a small amount of meta-level knowledge about how it is constructed. It first is trained on the meaning of its internal models. This is accomplished by associating the model name with the word(s) or phrases needed to properly orient to the named model. This is a many to one map where multiple words or phrases may isolate a single named internal model. Subsequently any of these models may be populated by being named either explicitly or implicitly. The trainer has to follow training decorum in order to be sure that the internal models are populated properly.

Subsequently, CR1 can detect the user communications context using the srModels if the phrase occurs exactly in the future. If the phrase is not an exact repetition, then the Links node produces the closest match with an associated binding. If the earned value is high and the binding cost is low, then the PDA may act on such partial information. If the earned value is not high enough or the binding cost is too high, the PDA will ask permission before acting.

7.3.2 Acquiring Radio Knowledge

One advantage of this somewhat shallow model of natural language processing is that it is directly applicable to radio protocols such as downloads and mode selection. RKRL data was used to train the system about the parameters of GSM, GPRS, and a notional RF LAN. Its internal models of modem, etc. were acquired using exactly the training machinery defined above. The trainer says:

"train GPRS dataRate CIR >30 is 21.4 done "

This sequence installs 21.4 as the dataRate model for a stimulus GPRS@CIR > 30 (dB). In CR1, that model was trained following the data content of the RKRL 0.3 XML. Automatic instantiation of RKRL 0.3 statements into future cognitive radio srModels would be a logical step in the further development of cognitive radio. The flow of information to the lateral models associated with this training sequence is as illustrated in Figure 7-16.

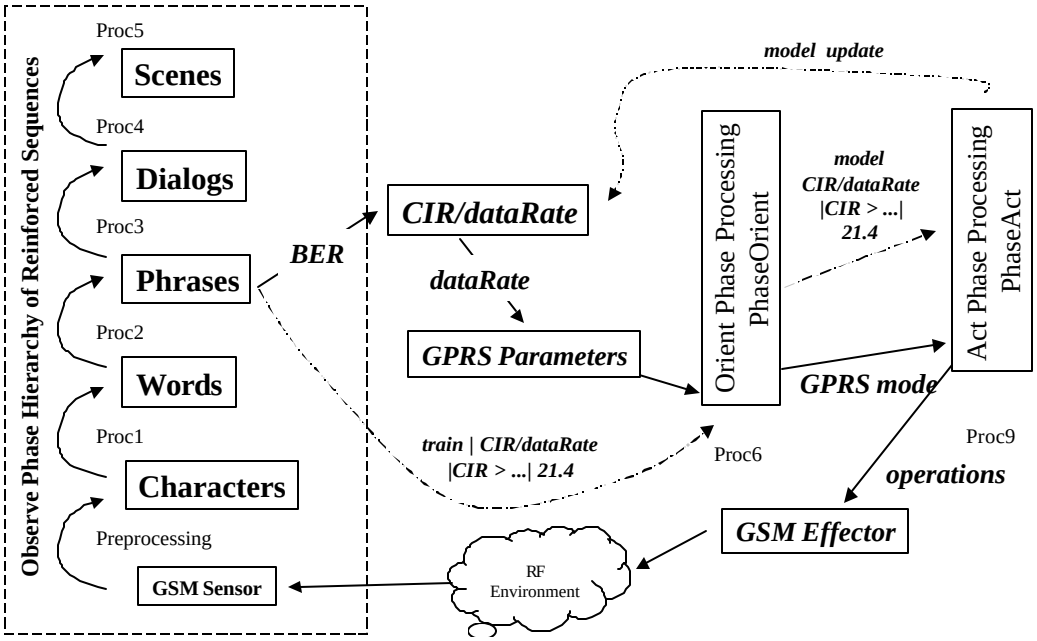


Figure 7-16 Acquiring and Using Radio Knowledge

During training, the sequence from the speech or text user interface (or from KQML via RF) is parsed, yielding phrase senses that update the CIR/dataRate model. Other models that participate in the setting of the GPRS mode as a function of context are also updated. Subsequently, the models process the BER as a function of context to set the appropriate GPRS mode. Since user context drives mode selection, it may choose a low data rate high probability of connection mode if the context so dictates. Thus, the processing flow within CR1 establishes the character of setting of radio bands and modes as a function of user communications context.

7.4 The Sleep Cycle

After a session of interaction with the user and the radio networks, the PDA requests a sleep epoch. During this period of time, stimuli are to be integrated into the PDA's set of knowledge. CR1's sleep cycle accomplishes the following tasks:

1. Transfers those items since the last sleep cycle from the novelty and accumulator-class nodes to the real-time performance nodes, the partner srNodes.
2. Replenishes the capacity of the novelty nodes and accumulators to acquire new knowledge, e.g. by clearing their srModels and resetting the capacity to an appropriate level.

3. Digests the models in the srNodes to assure that the computational capacity remains within bounds. The primary issue is to manage the memory of the ObjectSpace Hash Maps so that they are not more than 50% occupied, either by allocating more memory or by removing nodes with the lowest estimated value. Estimated value is a function that combines count, temporal extent, and context. For example, all srModel (stimulus, response) pairs with a trainButton in the context are preferred over those without that preferential context. (This aspect has been defined but not fully implemented in the rapid prototype because, unexpectedly, CR1 never ran out of memory.

In a more fully developed system, the PDA would reconcile negative reinforcement items via a machine learning process such as decision tree formation [51], In addition, CR1 would generate questions from the word, phrase, dialog, and scene-level srModels.

Given any new dialog, CR1 learns the dialog as both the stimulus and response of the the dialog level srModel. Because of this specific type of knowledge organization, CR1 could readily compose questions like, "What kind of dialog is 'Let me introduce Chip?'" A few such questions could be queued for user interaction upon wakening. If the user replies "Introductions," then the response to the srModel of 'Let me introduce Chip' would be set to "Introductions." Subsequently, whenever a matching expression is detected, the user-communications context "Introductions" would be reinforced to the Orient phase. This maps dialogs to scenes in a way that acquires this generalization knowledge relatively painlessly from a helpful user, incrementally customizing the system's communications services. Alternatively, a network administrator could provide the training for users that are not so helpful.

The transfer of srModels from novelty and accumulator nodes that acquire the raw learning data to the srModels that perform the associations in real-time acquires procedural knowledge. This is akin to encapsulation of skill through repetition. Fodor's monograph *Modularity Of Mind* [192] argues that such encapsulation is a cornerstone of human learning. The sleep cycle attempts to emulate that encapsulation algorithmically. R. Sun's CLARION system [193], on the other hand, has a two-layer learning architecture that learns high level declarative knowledge (predicates) from low level procedural knowledge (Q-learning). Such encapsulation is necessary for a resource-limited system that has to perform acceptably in real time (or die, perhaps, as in theories of human evolution). The architecture therefore accommodates a variety of such

approaches.

In a more fully developed system, the new terms would be analyzed for contextual cues to role. For example, the binding of a new word to a pattern in which the known word has a known word-sense can be interpreted as evidence that the new word shares the word sense with the known word. In the example discussed previously:

Layer3(Let me introduce chip, Let me introduce firstname) with firstname = chip.

In CR1, firstname has no particular word role, since its internal models are designed to detect communications context (introductions), not to populate databases. But the sleep cycle is designed to facilitate the integration of other machine learning algorithms. If the cognitive radio kept a natural language address book, the word-sense of firstname would be Name. Therefore, a set-cover or abductive inference algorithm could infer the word-sense of Chip as a Name from the context in which it was introduced. The architecture therefore supports the integration of additional machine learning algorithms as applications opportunities arise. In CR1, the sleep cycle refreshes the novelty nodes, transferring the new stimuli to the known srModels. It also reinforces the stimuli, increasing the reinforcement level of prior stimuli. A more elaborate sleep cycle could remove non-training examples that are not highly reinforced, gradually removing noise. CR1 therefore demonstrates a framework for machine learning beyond the capabilities implemented in this rapid prototype.

7.5 The Prayer Cycle

Not all items that cannot be resolved during the sleep cycle may be resolved through interaction with the user. The architecture envisions the resolution of the remaining items by interaction with external entities. These include the network operator(s), perhaps a cognitive radio home page, and, of course, the designer(s). The need for some such external support is clear. CR1 did not implement such a cycle, however. Instead, the Inspector was used to diagnose CR1's internal states. KQML interaction with a cognitive network seems particularly appropriate to a sleep cycle, especially since it is likely to occur in off-peak hours, given the simulations of the earlier use cases.

7.6 Performance Metrics

The primary performance metrics associated with cognitive radio would measure user satisfaction with the PDA's control of wireless voice, data, and/ or multimedia services. Quantification of these metrics can draw from established disciplines.

7.6.1 Precision and Recall of User Communications Context

Probability of detection (Pd) of a communications context is analogous to the recall metric of text retrieval/ document clustering. This is the number of documents retrieved expressed as a fraction of those that are known a-priori to be relevant. Similarly, CR1's ability to detect contexts for which it has been trained may be expressed as:

Pd = (Detections/ Events)

Probability of false alarm (P_{fa}) is similar to the precision metric of text retrieval. Precision is the number of documents that are relevant expressed as a fraction of the number retrieved. Thus, if 100 contexts are detected and 90 of them represent the state of the user, the precision is 90%. For large text corpora, algorithms that employ stemming, hierarchical clustering, and a certainty calculus achieve recall in the 80% range with precision of 50% [80]. Similarly, cognitive radio's false alarm rate would be measured as follows.

P_{fa} = (Detections-Errors)/Detections

An error is any detection that leads to negative reinforcement by the user.

7.6.2 Confusion Matrices

Although precision and recall characterize error rates, they do not provide much insight into the nature of the errors. The confusion matrix expresses the rate at which Context A is mistaken for Context B. The non-normalized confusion matrix of Table 7-4 shows a confusion matrix for 8 test cases in which CR1 was trained to detect introductions, eating (e.g. at home), and eating out. The test is not very representative because of the small amount of data available and because the test data was prepared by the developer. In a valid test, the training data and test data would be obtained from uncorrelated sources. As a minimum, the tester should be unaware of the training data and conversely. The confusion matrix brings rigor to the evaluation of performance, when that becomes the primary research objective. Since this thesis is oriented

towards architecture, substantial effort was not put into training and testing of performance, but into the analysis of the data structures and information flows supporting the reasoning architecture.

Table 7-4 Communications Event Confusion Matrix

↓ Actual/ PDA →	Introductions	Eating Out	Eating	Other
Introductions (2)	2	0	0	0
Eating Out(2)	0	1	1	
Eating(2)		1	1	
Other(2)		1		1

7.6.3 Conflict Rate

The rate at which the PDA's actions have to be overridden by the user is also a useful metric. The use concept behind cognitive radio is that the user can always pre-empt the PDA. Such preemption may be processed as a training experience. A follow-up dialog would extract from the user reasons for the conflict. This is a subject for future research in the machine learning of computational models of discourse.

7.6.4 Combinatorial Aspects

Finally, the use of memory and processing resources by cognitive radio's intelligent agents cannot be ignored. A software radio continues to emerge, the complexity of modulation, equalizer, coding, decoding, and the protocol stacks continues to increase. Interference suppression, for example, can enhance Carrier to Interference Ratio (CIR) for an increase in processing capacity. In some cases, the number of operations per bit can increase by three orders of magnitude between the most elementary conventional algorithms and advanced algorithms like sequential interference cancellation for W-CDMA [194]. In addition, trellis decoding is memory intensive. Thus, cognitive radio has to balance processing and memory resources of the software radio against other uses including user applications and cognitive processing. The following preliminary combinatorial analysis of the computational needs of cognitive processing is therefore relevant.

The most severe user of computational resources in CR1 is the observation phase. A fully

trained version of CR1 could have a 2000 word vocabulary with competence distributed among 50-100 models. The number of stock phrases such as introductions, asking for directions, etc. could be 2000. Trained contexts would be less because multiple phrases align to each context. The number of dialogs and scenes, similarly, are reduced as one climbs the abstraction hierarchy. The space complexity of reinforced hierarchical sequences is quadratic, proportional to the square of the number of items because of the correlation function of the links nodes. The linear factors due to the other nodes comprise less than the square root of the resulting total, so that is ignored in this analysis.

Table 7-5 Combinatorial Analysis of Space (Entries)

Aspect	#Known	Links
Trained Characters	40	0
Trained Words	2,000	80,000
Trained Phrases	2,000	4,000,000
Trained Contexts	1,000	2,000,000
Trained Dialogs	500	500,000
Trained Scenes	100	50,000

The space required is calculated by multiplying the number of elements (e.g. characters) times the number of known sequences (#Known, e.g. words), an almost quadratic function. The number of items to be stored in Hash Maps, then is about 6.63 million, plus or minus a factor of two for uncertainty in the tradeoffs of amount of training data versus performance. Since Hash Maps like 50% spare space to assure O(1) versus O(n) recall speed, space must be reserved for another 6.6 million items (to be conservative). If items require on the average 100 bytes of storage (pretty efficient, but not unattainable), then the PDA needs 1.3 GB of RAM. According to Moore's law, that amount of memory will be here within a decade in a PDA, so the memory-intensive approach required to support reinforced hierarchical sequences is not out of reach. In addition, the human brain is thought to contain on the order of 10^{15} elements of storage, so it is not out of the question that human reasoning may employ similar mechanisms. In fact, the human brain is known to be massively parallel and to store information in a distributed way. Thinking of the structure of the hash-maps in computer memory, one has a highly memory-intensive and massively parallel algorithmic architecture that might well map onto some special purpose processors. The clustering of stimulus-response knowledge in PDANodes partitions that knowledge for convenient mapping to massively parallel hardware, if one were so inclined.

The rate of growth of memory requirements will be a function of the degree to which the PDA is exposed to new stimuli. A PDA that is used "around the house" and not for extensive Internet queries, but perhaps for shopping will be exposed to a low rate of new stimuli. This could include twenty new words and 1000 new phrases per day of strong use (mostly from Internet queries while engaged in consumer practices such as shopping). It is hard to tell just how much memory would be needed to support meaningful use without forgetting so much that the consumer is dissatisfied. Examination of the prosodic correlation properties of Internet text would no doubt shed light on this aspect of cognitive radio.

Finally, the above analysis emphasizes human communication because that is the most demanding in terms of generality of expression and complexity of storage. The radio knowledge consisting of 4000 essentially phrase-level chunks of knowledge need not be stored in every PDA. The PDA's a-priori training is assumed to include the 1000 elements related to the PDA's network, area of sale and areas of the world in which the PDA would normally operate. The number of radio words is extremely low (< 100) and the number of radio contexts is also very low (<100) On the other hand, if one were to use cognitive radio to map propagation environment, then the storage required increases linearly with the number of locations known. Assuming 100 meter GPS regions, all the propagation knowledge of a user's track in the urban scenario would take less than 1 million elements of storage, less than 20% of the notional capacity of the PDA. An interesting question, then, arises of how much can one afford to reallocate memory to propagation measurement to support a network or user who is very concerned about radio performance (e.g. a military user) versus how much memory one might like to have to support natural language processing.

7.7 Observations and Research Issues

This section summarizes the research issues and lessons that were learned from the development of CR1. Any inter-disciplinary research project such as this generates questions that are already at least partially under consideration in the separate disciplines. The goal of this section is to highlight the nature of the intersection of those disciplines with cognitive radio.

7.7.1 Word Vector Research Issue

The lower layers of the sequence-recognition structure for user context detection is similar

to word-vector document clustering [30]. Sahmi's information-theoretic normalization approach works well for documents because there are enough words for statistical significance. One could process dialogs obtained from simulated interactions with cognitive PDAs in the same way that Sahmi's algorithms process documents. The resulting word-vectors could be used in mid-layer processing to identify contexts at the level of dialogs and scenes, e.g. in the sleep cycle.

There are few words to cluster in real-time dialog interaction with a PDA. Thus, symbol, word, and sequence counts are not meaningful. For this reason, the element counts are not explicitly normalized in CR1. Implicit normalization occurs in the activation of the strongest high level sequence(s) in the sequence matcher. Consider: "PDA, make me some coffee" (PDA Action: Request coffee from home's RF LAN). Or, "How's the weather?" The PDA repeats latest sequence from NOAA (National Oceanographic and Atmospheric Administration) weather radio. The topic is reliably detected if the words in the current situation occur exactly in that order in prior experience in which the user trained the PDA by manually selecting the weather channel. Therefore, instead of normalizing the reference knowledge in a probabilistic way, the non-normalized activation strength is the degree of belief. This is tantamount to the weight of the equivalent word vector. In addition, partial words partially activate word nodes. This allows CR1 to link word variants that occur in similar phrase contexts. If that link is subsequently negatively reinforced, the link's strength is reduced (count is decremented by k) so that that sequence is no longer the strongest when multiple activations occur.

Statistically-oriented reinforcement normalization may be accomplished during a sleep epoch. Since CR1 assimilates all words, phrases, dialogs, and scenes in novelty nodes during a wake cycle, they may be readily accessed for batch processing during a sleep cycle. The metrics of the statistical learning algorithm should be re-normalized to the integer counts in order to be consistent with CR1's real-time reinforcement algorithms. Although probability theory pushes one in the direction of normalizing over [0, 1], CR1 needs normalization instead over the integer domain [1, Max]. Any stimulus that is part of a PDA's experience has been observed at least once (in training), so zero is meaningless as a reinforcement value. Zero might mean that the sequence is thought to be illegal, but it would not be a reinforcement value. Max is the maximum number of reinforcements observed of any stimulus in any srModel. Although some bias is introduced by approximating a probability by a scaled integer, small integers are easier to

interpret algorithmically than small fractions. For example, if the reinforcement value of a sequence is 1, then the sequence has been observed exactly once. If the reinforcements are normalized to 1 like a probability, then it is difficult to tell what $3.23*10^{-6}$ means. Has that stimulus appeared only once or more often? A phrase that has appeared only once in the past year might be more safely purged than one that has appeared even twice. Consider the combinatorial limits to such an approach. The number of seconds in 10 years can be represented in a 29 bit integer. Thus, a 64 bit integer is capable of counting all the elementary sequences and their combinations presented to a 20-channel PDA in a 30 year life. Thus, the use of integer counts as degree of belief scales up to practical applications, subject to truncation errors.

7.7.2 Memory-based Learning

Knowledge representation in srModels follows case-based or memory-based learning. The use of memory in CR1 is therefore subject to combinatorial explosion. For example, only 18 phrases short translate into 569 srSequences in 25 srModels. The impact on real-time performance and memory use was less than feared. The Java prototype used the ObjectSpace HashMap and HashSet constructions for all internal models. These constructs save sequences efficiently and recall them in O(1) steps if the memory is only half full[12]. This means that phrases did not have to be pruned (in the sleep cycle) to stay within the bounds of a few hundred phrases readily supported on the Dell Laptop used for development. The laptop's 128 MB of RAM sufficed for near-real-time performance of the test cases.

In addition, memory-based learning exhibited high domain selectivity. For example, the cognitive radio was trained the exclusive-or relation that daunted linear perceptrons and that require many iterations of learning for a neural network. Memory-based learning assimilates the relation in exactly one exposure. It can tell you what 1 XOR 1 is because that stimulus binds so precisely to the case 1 XOR 1 equals 0.

The ability to train the PDA using sequences seemed natural. One of the touted advantages of memory-based learning is that it requires no a-priori intensional model of the domain. Indeed,

12 O(1) is a fixed number of steps independent of the size of the hash map; O(N) is linear growth; $O(2^N)$ is exponential growth, etc.

one can express anything from the Berlitz phrase book, from the SDR Forum download protocol, and from the GSM API without recourse to an a-priori model. This gives a lot of latitude in defining (and re-defining) the details of the knowledge representation. This is also consistent with the need for RKRL to support heterogeneous representations. Sequence matching is a simple, computationally efficient way of implementing the functional equivalent of rule-binding or resolution theorem proving over small rule sets.

This does not mean that hierarchical sequences are "better" than expert rules or Prolog. In general domains, one needs a Turing-capable engine with the power of Prolog. Prolog is much better than hierarchical sequences for general inference tasks. But cognitive radio is not a general reasoning domain. Its functions are limited to the identification of user communications contexts and the support of wireless access. But these functions have to be performed reliably in short, isochronous windows, either for the user or for the network. In addition, the resources to be used have to be predicted in advance of their use (e.g. in the planning phase). Thus, the advantages seem to outweigh the lack of generality. Uniformity of expression of words, phrases, dialogs, and scenes in testing and training, and across speech, text, radio protocols, etc. made the system easy to train and evaluate.

The primary research issue associated with memory-based learning in cognitive radio might be generalization. Memory-based learning by itself offers few strategies for generating generalized templates from a large collection of nearly identical templates. Otherwise, the consolidation of phrase and dialog level interactions into coherent scenes is left to an interaction with the user. Perhaps the combination of text-retrieval approaches and memory-based approaches would be an interesting research area. In addition, one would like to be able to replace a long dialog with a user about, say, sending email with a shorter sequence that accomplishes the same result and that frustrates the user less. This is akin to the search for macro-operators, possibly using means-ends analysis [54, 28].

7.7.3 Difficulties in Quantitative Assessments of Interactive Agents

How does one develop the large sets of data needed to generate standard confusion matrices? In TREC [80], for example, there are large standard corpora of text data. During the mid-1980's numerous "good" text retrieval engines were developed and tested on standard text

corpora of about 60 MB. Later, the first 2 GB corpus showed how truly deficient these "good" algorithms were. The situation is similar for CR1. CR1's "corpus" of dialogs would depend on the way CR1 interacts with the environment and user. This makes it a very labor-intensive, time-consuming task to create and run meaningful scenarios. What techniques could be used to create large, standard data sets? Maguire's notion of using Berlitz language guides for training might be extended. One could identify language training scenarios for training the user-interface processing of a cognitive PDAs. One might then use standard foreign language tests in which a small but representative number of alternatives can be generated, e.g. by a state machine simulation of a user.

How does one characterize the relationship between the closed set of knowledge that can be sustained inside of a PDA and the open-set of knowledge in the world? The digital libraries program showed that multiplicative normalization and hierarchical document partitioning yields better scores than non-hierarchical approaches. RKRL provides a reference taxonomy that might be used in getting at this issue. Obvious metrics like the size of the PDA's vocabulary and the number and scope of its acquired models might provide a superficial treatment. But the equivalent of a confusion matrix is hard to formulate for open-set constructs.

7.7.4 Grounding versus Difference of Opinion

How does one characterize the degree of success of human-computer communication? Suppose the user asks the PDA: "Will it rain today?" The PDA plays back the latest NOAA weather broadcast. It has inferred "Rain" as its weather state. The user says, "No, I do not think it will rain today." Has the PDA successfully communicated with the user? This aspect of human-computer communication is often called grounding. Grounding is the alignment of internal states of two cognitive entities. Grounding errors can include (a) failure of one entity to express an internal state to another, (b) lack of access of the recipient to the expression, (c) and misinterpretation of a perceived expression. A metric has been described which expresses an agent's inclination towards a communications act to correct a detected grounding error [195]. A state-confusion matrix (patterned after the confusion matrices used in pattern recognition and language processing) could compare the internal states of two agents that are supposed to be sharing identical internal states. Such a matrix would characterize the success of

communications, but would not give much insight into the underlying agent-agent or human-machine communications phenomenology. Additional metrics might:

a. Express the relative importance of internal states with respect to the task at hand.

b. Express the agent's ability to know its own internal state. Most internal states of conventional software radios are not computationally accessible. Although many internal states of cognitive radios are accessible, many meta-level states are not. For example, CR1 does not yet have algorithms that diagnose its need for a new srModel. Thus, it can not express such a need. A predicate over meta states might be used to quantify whether agents have the capabilities required for grounding.

c. Express the reliability of one agent's means of expressing internal states to another agent. That is, it would express the degree to which two agents share language syntax and semantics.

d. Express the cost of expressing the internal state. Some states might be expressed only through long dialogs, or at the expense of doing other tasks. How can such costs be quantified?

Some function could map these additional metrics into a space suitable for comparing agents' communications capabilities. Alternatively, items a-d could be assessed as independent dimensions of agent communications.

Various forms of disagreement have been treated as grounding errors, but that view seems misleading. If the PDA's internal state reflects X ("It will rain today"), but the user thinks !X (not X, e.g. "It will not rain today"), the two disagree about the weather. If a cognitive PDA's internal state was derived from the speech processing of the NOAA radio weather broadcast, its internal state was derived from a legitimate source. The user's view may be based on experience that the weather forecasters are wrong 30% of the time, perhaps coupled with optimism. If the PDA expresses its internal state to the user, then miscommunication has not occurred. If the user tells the PDA "It will not rain today," then the PDA should detect a conflict. Suppose it looks up its srModel for weather and finds "Today -> Rain." This was deposited there by its processing of the NOAA weather forecast. It can say to the user "NOAA Weather Radio says it will rain today." The reply might be "I don't think so." At which point, the PDA could replace "Today -

> Rain" with "Today -> No Rain". Alternatively, it could extend the context by creating two pairs: "(NOAA, Today) -> Rain" and "(User, Today)->No Rain" in its srModel of Weather. In either case, the user and the PDA are communicating about a state where the differences are not due to miscommunication. The extended context approach resolves the conflict of internal states by augmenting the internal state in such a way that the PDA's problem of storing conflicting models is resolved. The disagreement remains, but the computational difficulty is overcome. In addition, the PDA can employ the user's state ("no rain") when acting on behalf of the user (e.g. reserve a table on the patio at the restaurant).

The observation is that conflict presents an opportunity for a cognitive agent to learn. In order to do this in a computationally-effective way, the agent must:

(a) Detect the conflict (e.g. replacing "Rain" with "No Rain" is a problem),

(b) Encode the situation that led to the conflict as a problem state (User versus NOAA as sources for the srModel of weather),

(c) Recognize the aspect of context that can convert a conflict over internal state into multiple internal states (extend the Weather model stimulus from Today, a temporal constant to (User, Today), a vector), and

(d) Obtain the data needed to create multiple states (in the example, srWeather: User -> "Today-> No Rain" and srWeather: NOAA -> "Today -> Rain" existed internally, but would not be recognized as such without specialized internal data structures and related processing), and finally

(e) Deal with the multiple internal states from that point forward. It would update all of its internal models that ask about weather to express (agent, time) as the stimulus to the srWeather model. It could also install the dialog "Today -> ERROR | Askfor (<agent>, Today)" in the srWeather model. Any internal or external process that asks for the weather will know that that is a function of the context variable <agent>.

(f) Setting up and accessing such context variables imparts implies internal structure to the cognitive agent that may not be present until such a conflict is detected. Mechanisms to do this are open-ended in that the PDA can reshape its internal structure, potentially

leading to unstable behavior (such a PDA would be Turing-capable and thus potentially unstable)

A disagreement regarding the value of an internal state is a specific example of conflict. Other types of conflict are present in CR1. For example, a user can always take over control of the PDA's radio resources in a way that differs from the plan offered to the user by the PDA. In any of these situations, the PDA is presented with a learning opportunity. The reinforcement mechanism built into CR1 could trigger communications with the user, to employ internal machine-learning algorithms, and to allocate more internal resources (e.g. permanent case memory) to those situations that occur often enough. Those conflicts that occur only once might be safely ignored. Conflicts occurring more than once might be dealt with through machine learning in the next sleep cycle. Those that cannot be resolved autonomously could be presented to the network or to the user for assistance. Only a very limited implementation of this machine learning strategy has been implemented in CR1.

The use cases, RKRL, the design of CR1, and these observations support the definition of an architecture for cognitive radio, presented in the next chapter.

8 The Cognitive Radio Architecture

An architecture for cognitive radio consists of the functions, components, and design rules necessary to support the evolution of cognitive radio. The architecture integrates the contributions of researchers focusing on the specific disciplines of software radio, network engineering, natural language processing, and machine learning. This architecture minimizes the dependence on knowledge-engineering through the integration of machine learning. It also minimizes the burden on the user through the integration of natural language processing. This chapter defines the cognitive radio architecture that emerged through the definition of RKRL and its refinement and use in CR1.

8.1 Primary Cognitive Radio Functions

Cognitive radio is a software-defined radio that incorporates applications, interfaces, and cognition functions identified in Figure 8-1.

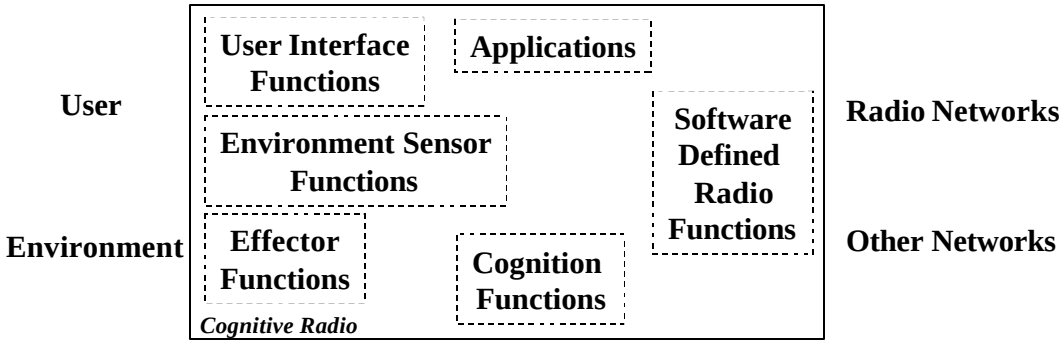


Figure 8-1 Functions of Cognitive Radio

The primary radio cognition functions consist of:

1. Recognize user communications context.
2. Mediate wireless information services as a function of context.

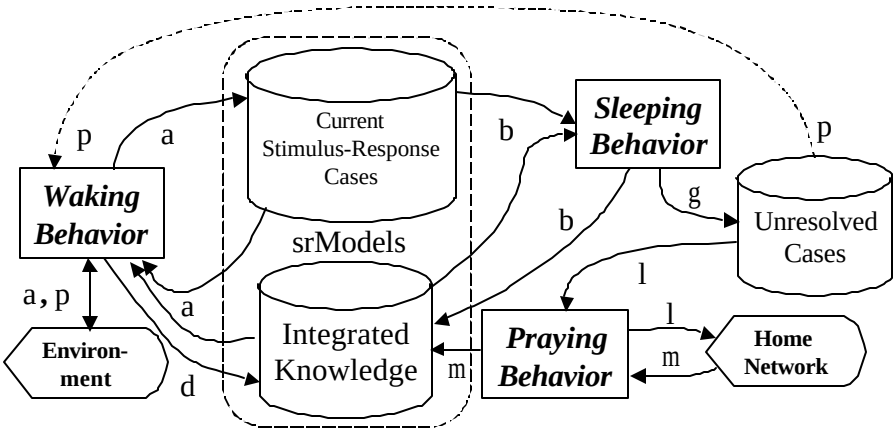
The recognition of user communications contexts should be passive to the maximum extent possible. That is, the cognition functions should rely on processing streams of user interaction

with applications as its primary means of inferring communications context. Only as a last resort should the cognition functions require user input that has no purpose other than to assist the cognition functions. When such input is necessary, the cognition functions should structure that interaction to be as intuitively appealing as possible.

The mediation of wireless information services includes continually keeping track of the parameters of wireless networks are present in the environment. These parameters include spectrum occupancy, received signal strength as a function of time and space, available QoS, and related cost(s). The fidelity of parameter estimation should be sufficient to plan context-sensitive wireless access on behalf of the user.

8.2 Behaviors

Cognitive radio support functions include three modes of behavior: waking, sleeping, and praying. Behavior that lasts for a specific time interval is called a behavioral epoch. The axiomatic relationships among these behaviors are expressed in the topological maps of Figure 8-2.



a - **action cycle;** d - **incremental machine learning;**
b - **non-incremental machine learning;** g - **learning conflicts**
l - **RKRL/KQML requests for external assistance;**
m - **authoritative assistance;** p - **attempt to resolve problems**

Figure 8-2 Cognitive Behavior Model Consists of Domains and Topological Maps

8.2.1 Waking Behavior

The waking behavior is optimized for real-time interaction with the user, isochronous control of software radio assets, and real-time sensing of the environment. The conduct of the

waking behavior is informally referred to as the awake-state, although it is not a specific system state, but a set of behaviors. Thus, the awake-state cognition (α) maps environment interactions to the current stimulus-response cases, the dynamic subset of the embedded srModels. Incremental machine learning (δ) maps these interactions to integrated knowledge, the persistent subset of the srModels.

8.2.2 Sleeping Behavior

Cognitive PDAs detect conditions that permit or require sleep. For example, if the PDA predicts or becomes aware of a long epoch of disuse (e.g. overnight), then it PDA may autonomously initiate sleeping behavior. Otherwise, it would request permission to enter sleeping behavior from the user. During the sleep epoch, the PDA process experience from the waking behavior using non-incremental machine-learning algorithms. These algorithms map current cases and integrated knowledge onto integrated knowledge (β).

A conflict is a context where the user overrode a PDA decision about which the PDA had no little or no uncertainty. Map β may resolve the conflict. If not, then it will place the conflict on a list of unresolved conflicts (map γ).

8.2.3 Prayer Behavior

Attempts to resolve unresolved conflicts via the mediation of the PDA's home network may be called prayer behavior. The unresolved-conflicts-list is mapped (λ) to RKRL XML queries to the PDA's home network expressed in KQML. Successful resolution maps network responses to integrated knowledge (μ). Alternatively, the PDA may present the conflict sequence to the user, requesting the user's advice during the wake cycle (map π).

8.3 Cognitive Radio Components

Cognition functions implemented via cognition components. These include data structures and related processing components identified in Figure 8-3. The cognition functions are allocated to these components. Data structures are shown as rectangles (e.g. Dialogs), while processing components are shown as maps (e.g. P). The important entities are shown as rounded rectangles. These are the world, W, the PDA, and the PDA's World Model, S. Entities in the world include the differentiated entities "Own User" and "Home Network." The

architecture requires that the PDA be able to identify these entities so that it may treat them differentially. Other networks, people, places, and things may be identified in support of the primary cognition functions, but the architecture does not depend on such a capability.

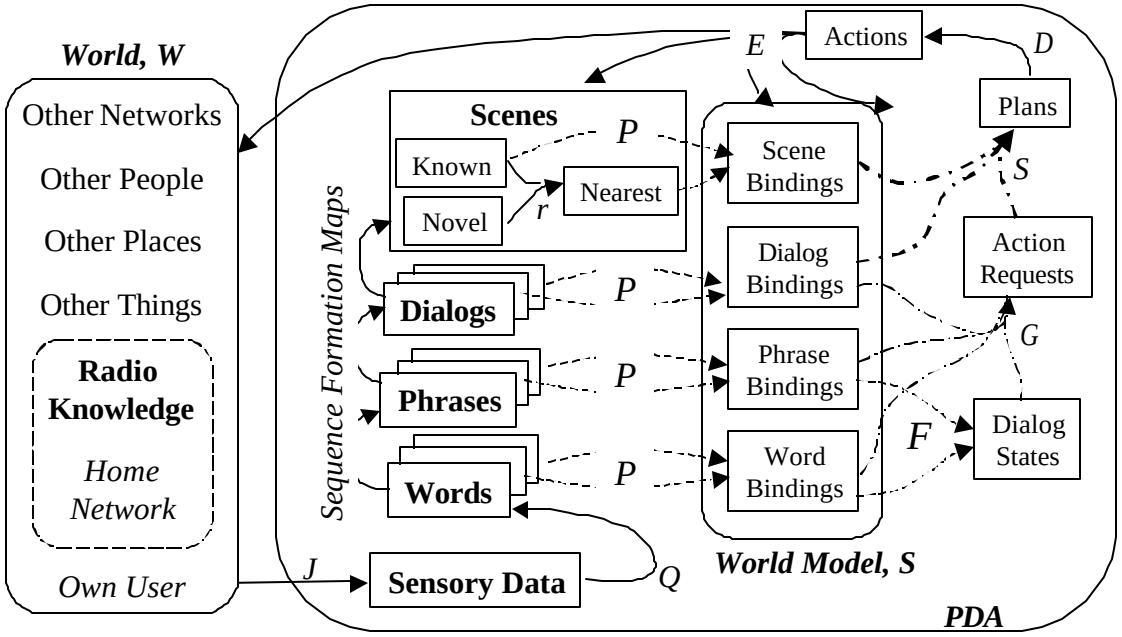


Figure 8-3 Architecture Based on The Cognition Cycle

The data structures include the reinforced hierarchical sequences Words, Phrases, Dialogs, and Scenes of the Observe Phase. Within each of these sequences, the novel sequences represent the current stimulus-response cases of the cognitive behavior model. The known sequences represent the integrated knowledge of the cognitive behavior model. Known sequences may consist of RKRL statements embedded in the PDA or of knowledge acquired through independent machine learning. The Nearest sequence is the known sequence that is closest in some sense to the novel sequence. The ***World Model, S***, consists primarily of bindings between a-priori data structures and the current scene. These structures are also associated with the Observe Phase. Dialog states, action requests, plans, and actions are additional data structures needed for the Observe, Orient, Plan, and Act Phases respectively. Each internal data structure maps to an RKRL frame consisting of handle (e.g. set, or stimulus), model, body (e.g. subset, or response), resources, and context (RKRL URL, source, time, and place of the frame).

The processing components of the architecture are modeled as topological maps, following the approach defined in Appendix B. The input map *J* consists of components that transform

external stimuli to the internal data structure Sensory Data. The transformation Q consists of speech recognition, lower-level software radio waveform interface components, etc., that create streams of primitive reinforced sequences. A cognitive radio also includes components that form successively higher level sequences from the data on the immediately lower level. Reasoning components include the map r that identifies the best match of known sequences to novel sequences. These are bound to scene variable by projection components, P. The maps J, Q, r, and P constitute Observe Phase processing. Word and phrase level bindings are interpreted by the components F to form dialog states. Train, for example, is the dialog-state of a training experience in CR1. The components G create action requests from bindings and dialog states. The maps F and G constitute Orient Phase processing. Scene bindings include user communications context. Context-sensitive plans are created by the component S that evaluates action requests in the Plan Phase. The Decision Phase processing consists of map D that maps plans and scene context to actions. Finally, the map E consists of the effector components that change the PDA's internal states, change displays, synthesize speech, and transmit information on wireless networks using the software radio personalities. Selected aspects of this architecture are now elaborated.

8.4 A-Priori Knowledge Taxonomy

Cognitive radio is based on an a-priori taxonomy of knowledge summarized in Figure 8-4.

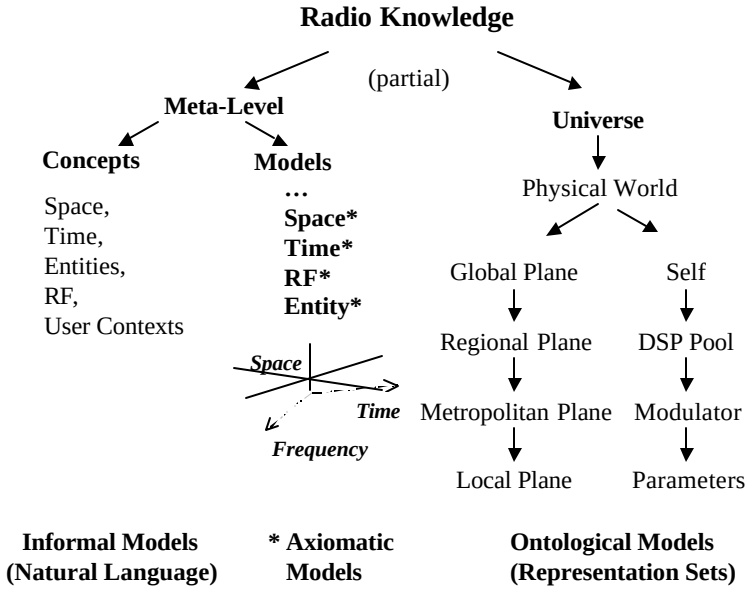


Figure 8-4 Minimum Taxonomy of Radio Knowledge

As a minimum, that taxonomy should include the named sets of RKRL 0.3. The radio knowledge should be axiomatized according to Chapter 6. Minimum concepts to be represented internal to the cognitive PDA include time, space, RF propagation, radio networks, external entities, and user communications contexts. In addition, the a-priori knowledge taxonomy must include a hierarchical model of the physical world and a model of the internal structure of the SDR.

8.5 Observe-Phase Data Structures

The observe-phase employs the reinforced hierarchical sequences of Figure 8-5. Word, phrase, dialog and scene should be topological spaces with the power-set topology.

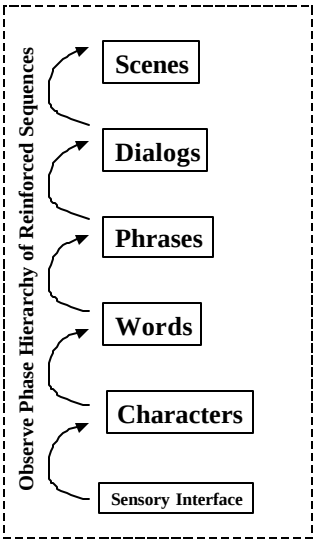


Figure 8-5 Reinforced Hierarchical Sequences

The components of the observation phase maps may be tailored. As illustrated in Figure 8-6, the speech and/or text channels may be processed via natural language facilities with substantial a-priori models of language and discourse. The use of those models should entail the use of homeomorphic mappings among the word, phrase, dialog, and scene levels of the observation phase hierarchy and the encapsulated component(s).

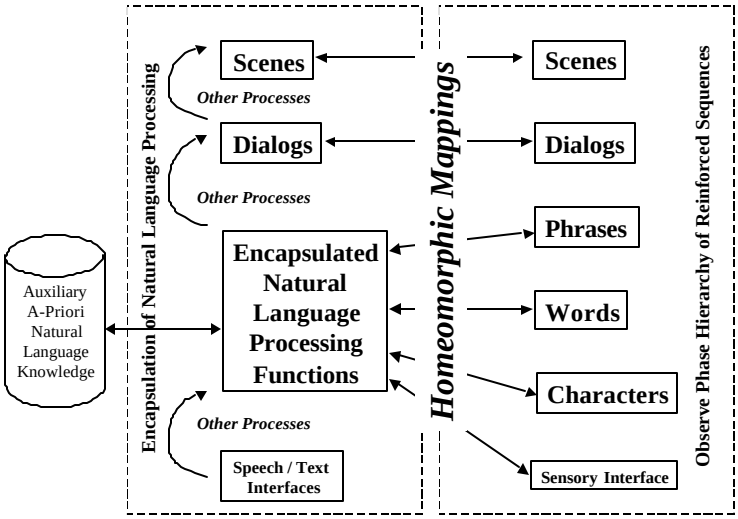


Figure 8-6 Natural Language Encapsulation

The Observe Phase components match new stimuli to known stimuli. When an exact match is not possible, the components may deliver one or more hypotheses. Hypotheses may consist of best-match, or a prioritized list of partial matches. Bindings may be computed as the interface from the observation phase hierarchy to the orient phase.

8.6 Radio Procedure Knowledge Encapsulation

Radio knowledge may be embodied in components called radio knowledge sources. If so, they are organized as set-theoretic maps among wake-cycle phases (observe, orient, plan, decide, act). A subset of these (e.g. radio procedure KSs) may control radio personalities as illustrated in Figure 8-7.

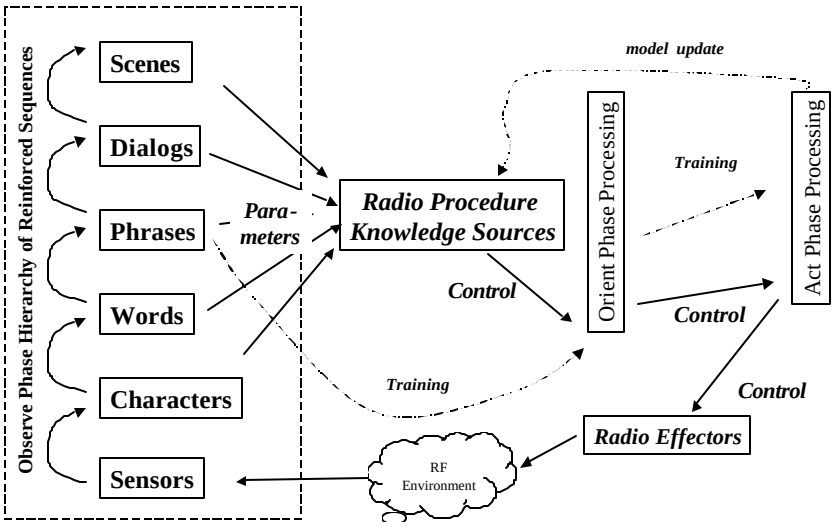


Figure 8-7 Radio Knowledge Sources Respond to Observations

These knowledge sources accept a-priori knowledge from RKRL an associated XML micro-world. Each knowledge source also saves the knowledge it learns in RKRL format, conforming to RKRL axioms.

8.7 Orient-phase Components

The Orient-phase determines the novelty of stimuli represented in the sequence hierarchy. The orient phase components bind variables and structure requests for planning. These components may directly invoke decide-phase or act phase components if the immediate context dictates. The orient-phase components may preempt planning if the stimuli strongly invoke immediate action. The orient phase determines when the PDA needs sleep or prayer behaviors and requests them according to the communications context.

8.8 Plan Phase Components

The plan phase represents plans for the control of the software radio personalities. The plan phase may include a plan calculus. The plan calculus may represent the following planning information:

<Plan> := <Plan-Phrase>| <Plan-Phrase>*

<Plan-Phrase> := <Request> <When> <Where> <What> |

<Request> <When> <Where> <Context><What>

<Request> := Say | Send | GoTo | Wait |Dial | Connect

<When> := Now | <Day>|<Month>|<Year>|<Hour>|<Minute>|<Second>|<DateTimeGroup>

<Where> := Here | <Place>|<Lat><Lon>|<X><Y>

<Context> := Any | Normal | Unknown | Introductions | Home (Inside) | Home (Outside) |

Commuting | Travelling | Hotel | Eating out | Sightseeing | Friends | Relaxing |

Shopping | Money | Medical | Other | <New-Context>

The plan calculus may include reasoning over time, space, user identities, and contexts. Each request corresponds to an operation that can be accomplished by an effector. The axiomatization of time permits times to be compared and time delays to be computed. The plan calculus generates and compares plans and detects conflicts. The Plan-phase generates alternatives, including expressing plans to peers and/or the network to obtain advice. A plan element may be a sequence that has been mapped to [Here, Now]. There may be competing and

partially fulfilled plans.

8.9 Decide-Phase Components

The decide-phase selects among alternatives generated by the planning phase. Its knowledge representation depends on the radio procedure components. The decide-phase allocates computational and radio-resources to subordinate (conventional radio) software, based on the activation of a plan. Autonomous decisions may be taken when the correlation between the current situation and the reference situation is high, and when the radio has been in this exact spatial-temporal context before. Tentative decisions may be offered to the user for confirmation, which makes it more likely to be invoked in the future. In the case of unresolved conflicts among plans, the decide phase lets the user do the task, observing and recording the user's initiation of low level communications and information processing tasks.

8.10 Act-Phase Knowledge Representation

The Act-stage initiates tasks with specified resources for specified amounts of time. The PDA thus provides the user with access to radio resources (bands, modes). The knowledge representation of the act phase depends on the radio knowledge sources contributing to the parameters of or participating in the action.

8.11 Design Rules

The following design rules circumscribe the way cognitive radio functions are mapped to the components of a wireless PDA within the envisioned architecture, as follows:

1. The cognition functions shall maintain an explicit topological model of the world, of the environment (including the user, the physical environment, and the radio networks, and of the internal states of the radio). Context shall be axiomatically represented using the topological model.
2. The model of the world shall follow an explicit axiomatic treatment of time, space, radio frequency, radio propagation, and the identity of entities.
3. Models shall be represented in an open architecture radio knowledge representation language suited to the representation of radio knowledge (e.g. RKRL 0.3). That language shall support topological properties and axiomatic models of cognitive radio.

4. The cognition functions shall maintain location awareness, including the sensing of location from global positioning satellites, and sensing position from local wireless sensors and networks. Location shall be an element of context.

5. The cognition functions shall maintain awareness of time to the accuracy necessary to support the control of radio functions. Time shall be an element of context.

6. The cognition functions shall maintain an awareness of the identity of the PDA, of its primary user, and of other legitimate users designated by the primary user. Current user shall be an element of context.

7. The cognition functions shall reliably infer the user's communications context and apply that knowledge to the provisioning of wireless access by the SDR function.

8. The cognition functions shall model the propagation of its own radio signals with sufficient fidelity to estimate interference to other spectrum users. The cognition function shall also assure that interference is within limits specified by the spectrum use protocols in effect in its location (e.g. in spectrum rental protocols). It shall defer to the wireless network in contexts where the network manages interference.

9. The cognition functions shall model the domain of applications running on the host platform, sufficient to infer the parameters needed to support the application. Parameters modeled include QoS, data rate, probability of link closure (Grade of Service), and space x time x context domain within which wireless support is needed.

10. The cognition functions shall configure and manage the SDR assets to include hardware resources, software personalities, and functional capabilities as a function of network constraints and use context.

11. The cognition functions shall administer the computational resources of the platform. The management of software radio resources may be delegated to an appropriate SDR function (e.g. the SDR Forum domain manager). Constraints and parameters of those SDR assets shall be modeled by the cognition functions. The cognition functions shall assure that the computational resources allocated to applications, interfaces, cognition and SDR functions are consistent with the user communications context.

12. The cognition functions shall represent degree of belief in external stimuli and in inferences. A certainty calculus shall be employed consistently in reasoning about uncertain information.

13. The cognition functions shall recognize preemptive actions taken by the network and/or the user. In case of conflict, the cognition functions shall defer the control of applications, interfaces, and/or SDR assets to the network or to the primary user, according to an appropriate operations assurance protocol.

9 Conclusions

This thesis introduces cognitive radio. It describes RKRL and summarizes CR1. Lessons learned from CR1 lead to the Cognitive Radio Architecture. The results from CR1 justify the conclusion that cognitive radio provides an interesting framework the integration of natural language processing and machine learning with software radio.

As the degree of computational intelligence of networks and handsets continues to increase, the allocation of intelligence between them takes on increased importance. Within 5 years, the capabilities of DSP's and memory will be such that the focus of this tradeoff will shift from what can be done (in terms of power limitations) to what should be done. Professor Maguire [26] has argued that the computing platforms will become so powerful that the emphasis in mobile communications will shift to the realms of environment awareness and ***mobile*** computer-to-computer communications. Cognitive radio is a computer-intensive technology to balance a user's communications and computing goals against those of a variety of networks with which that user could operate. This tradeoff includes the degree of flexibility that we create a-priori (e.g. via a new protocol) versus the degree of autonomy that we allow the radios themselves. The doctoral research explored this question through the refinement of RKRL into an XML baseline, and through the implementation and analysis of the Java rapid-prototype CR1.

9.1 Summary

The CR1 prototype includes a Java implementation of a large enough subset of the micro-worlds to characterize the evolutionary path of cognitive radio. The prototype developed in a simulated environment employs use-cases that examine specific aspects central to the use of cognitive radio in future wireless networks. The radio applications focus of the prototype complements the natural language processing and machine learning aspects that organize radio domain knowledge for the cognition cycle. This knowledge organization includes the use of RKRL for a-priori knowledge that can be shared via XML. It also includes the wake, sleep, and prayer behaviors. Supporting the wake behavior are the phases of the cognition cycle. The

reinforced hierarchical sequences of the observation phase structure both interactions with users and machine-oriented protocols such as secure downloads. The orient-, plan-, decide-, and act-phases each accommodate a different style of automated reasoning. CR1 established that it is possible to accomplish the functional needs of cognitive radio using these phases.

When the author presented cognitive radio as a potential novel approach to the management of the radio spectrum before the US Federal Communications Commission in April, 1999, he introduced the term cognitive radio to the media. Subsequently, the term cognitive radio has been used in the spectrum management community as a "really smart" radio. This is in spite of the fact that the definition supplied with the term stipulated that the radio employ model-based reasoning about its environment, location, radio propagation, networks, protocols, user, and its own internal structure. The more precise definitions based on point-set topology with associated axiomatic treatments should help serious researchers differentiate among what will no doubt become a proliferation of different types of very smart radios.

9.2 Research Issues

Maguire's question: "How do cognitive radios learn best?" merits attention. The exploration of learning in cognitive radio includes both the internal tuning of parameters and the external structuring of the environment to enhance machine learning. Since many aspects of wireless networks are artificial, they may be adjusted to enhance machine learning. This thesis did not attempt to answer these questions, but it frames them for future research.

This thesis therefore provides an important inter-disciplinary contribution linking natural language processing, machine learning, and radio engineering in a way that is useful for the further evolution of software radio.

10 References

1 J. Mitola III, *Software Radio: Wireless Architectures for the 21st Century* (Fairfax, VA: Mitola's STATIS*faction, Alpha Edition* - ISBN 0-9671233-0-5) 1999. The full text is being published under contract to Wiley Interscience, to be available in the year 2000.

2 J. Mitola III, *Cognitive Radio: Model-Based Competence for Software Radio*, Licentiate Thesis TRITA-IT AUH 99:04 (Stockholm, Sweden: KTH, The Royal Institute of Technology) August, 1999

3 J. Mitola III, *"Cognitive Radio: Agent-based Control of Software Radios",* Proceedings of the 1. Karlsruhe Workshop on Software Radio, 29 March 2000

4 J. Mitola III, "Cognitive Radio for Flexible Mobile Multimedia Communications" submitted to the IEEE Conference on Mobile Multimedia Communications (MOMUC) 1999, and Appendix D of this thesis.

5 ITU, Specification and Description Language, Recommendations Z.100 (Geneva: ITU) 1991

6 Methods for Testing and Specification (MTS); Strategy for the use of formal SDL for descriptive purposes in ETSI products, TR 101 081 V1.1.1 (Sophia Antipolis Cedex, FR: ETSI) 1997

7 Pesonen, "Object-Based Design of Embedded Software Using Real-Time Operating Systems" IEEE 1068-3070/94 (NY: IEEE Press, 1994)

8 J. Ellis, Objectifying Real Time Systems (NY: SIGS Books, 1994)

9 K. Ellison, Developing Real-Time Embedded Software in a Market-Driven Company (NY: Wiley, 1994)

10 Simpson, "The Duel Over Object Models" Client/Server Today, Aug 94

11 Eriksson and Penker, *UML Toolkit* (NY: John Wiley & Sons, Inc.) 1998

12 Mouly & Pautet, "Evolution of the GSM System" IEEE PCS Magazine (NY: IEEE, Oct 1995).

13 SDR FORUM TECHNICAL REPORT 2.0 Architecture and Elements of Software Defined Radio Systems as Related to Standards (Rome New York : SDR Forum) June 1999.

14 Erceg, "Comparisons of a Computer-Based Propagation Prediction Tool with Experimental Data Collected in

Urban Microcellular Environments" JSAC May 97

15 T. Mowbray and R. Zehavi, *The Essential CORBA,* Object Management Group (NY: Wiley) 1995

16 P. Rust et al, CML-Chemical Markup Language Poster (http://www.ch.ic.ac.uk/cml/) August 95

17 M. Adams, "Domain Profile Specification (Submission) for Mobile Working Software Architecture 1.2A" *Software Defined Radio Forum Contribution* : (Rome, NY: SDR Forum) Feb 00

18 J. Mitola III, "Software Radio: Technology and Prognosis" Proceedings of the IEEE National Telesystems Conference (NY: IEEE Press) May 1992

19 R. Walden, " Analog-To-Digital Converter Survey & Analysis" JSAC (NY: IEEE Press) Apr 99

20 V. Bose et al, "Virtual Radio" IEEE JSAC on Software Radios, (NY: IEEE Press) 1998

21 M. Cummings et al, "MMITS Standard Focuses on APIs and Download Features," (Cleveland, OH: Wireless Systems Design) April 1998

22 R. Hennie, *Introduction to Computability* (Reading, MA: Addison-Wesley) 1997

23 J. Meggers et al, "Mobile Multimedia for Small Handheld Devices", Proceedings of the ACTS Mobile Summit, (Rhodes, Greece: The European Commission) June 98

24 B. Kreller, "A Mobile-Aware City Guide Application," Proceedings of the ACTS Mobile Summit, (Rhodes, Greece: The European Commission) June 98

25 Xiaohan Yu, "Mobile Multimedia Services Based on GSM HSCSD, " Proceedings of the ACTS Mobile Summit, (Rhodes, Greece: The European Commission) June 98

26 H. W. Beadle, G. Q. Maguire, , and M. T. Smith, *Environment Aware Computing and Communications Systems*, (www.kth.edu: Royal Institute of Technology (KTH)) 1998

27 R. Michalski, *Machine Learning* (Palo Alto, CA: Tioga Publishing Company) 1983

28 P. Langley, *Elements of Machine Learning* (San Francisco: Morgan-Kaufmann) 1996

29 T. Kohonen, *Self-Organization and Associative Memory* (Berlin: Springer-Verlag) 1984

30 M. Sahmi, Stanford PhD Dissertation (Palo Alto, CA: Stanford University) 1998

31 R. Michalski, “Learning from Observation: Conceptual Clustering” in *Machine Learning* (Palo Alto, CA: Tioga Publishing Company) 1983

32 N. Nilsson, *Learning Machines* (New York: McGraw Hill) 1978

33 T. Menzies “Applications of abduction: knowledge-level modelling” (International Journal of Human-Computer Studies) Sept 1996

34 J. Hobbs et al “Interpretation as abduction” (Artificial Intelligence) Oct. '93

35 R. Davis, *The Use of Meta-Level Knowledge in the Construction and Maintenance of Large Knowledge Bases*, PhD Dissertation, Stanford University, Palo Alto, CA, 1982

36 McClelland and Rumelhardt, *Explorations in Parallel Distributed Processing* (Cambridge: MIT Press) 1988

37 R. Sutton and A. Barto, *Reinforcement Learning* (Cambridge, MA: The MIT Press) 1999

38 R. Sun and T. Peterson, “Multi-agent reinforcement learning: weighting and partitioning” *Neural Networks 12 (1999)* (Amsterdam, The Netherlands: Elsevier) 1999

39 R. Aler et al, “Genetic Algorithms and Deductive-Inductive Learning: a Multi-strategy Approach” Proceedings of the International Conference on Machine Learning (Irvine, CA: Morgan Kaufman) 1998

40 C. Watkins and P. Dayan “Q-Learning” *Machine Learning* vol 8 1992

41 D. Bertsekas and J. Tsitsiklis *Neural-Dynamic Programming* (Athena Scientific) 1996

42 H. Tong and T. Brown, “Adaptive Call Admission Control Under Quality of Service Constraints: A Reinforcement Learning Solution” IEEE JSAC (NY: IEEE Press) February 2000

43 L. M. G. Goncalves, “Multi-mode Stereognosis” Autonomous Agents 99 (Seatle, WA: ACM) 1999

44 R. Sun and C. Sessions, “Self-Segmentation of Sequences: Automatic Formation of Hierarchies of Sequential Behaviors” *IEEE Transactions on Systems, Man, and Cybernetics* (NY: IEEE Press) June 2000

45 R. Michalski, I. Bratko, and M. Kubat, *Machine Learning and Data Miningy* (NY: John Wiley & Sons, LTD) 1998

46 C. Ziemer and R. Peterson, Digital Communications and Spread Spectrum Systems (New York: Macmillan) 1985

47 R. Leopold, “The Iridium Communications Systems” Proceedings of ICCS/ ISTA (NY: IEEE Press) 1992

48 M. Mango and E. Auriol “Application of Inductive Learning and Case-Based Reasoning for Troubleshooting Industrial Machines” in R. Michalski et al editors, *Machine Learning and Data Miningy* (NY: John Wiley & Sons, LTD) 1998

49 J. O. Kephart, “Biologically Inspired Defences against Computer Viruses” ” in R. Michalski et al editors, *Machine Learning and Data Miningy* (NY: John Wiley & Sons, LTD) 1998

50 Bratko et al, “Applications of Inductive Logic Programming” in R. Michalski, I. Bratko, and M. Kubat, *Machine Learning and Data Mining* (NY: John Wiley & Sons, LTD) 1998

51 J. R. Quinlan, “Learning and Efficient Classification Procedures and their Application to Chess End Games” *Machine Learning* (Palo Alto, CA: Tioga Publishing Company) 1983

52 I. Watson, *Applying Case-Based Reasoning* (San Francisco, CA: Morgan-Kaufman) 1997

53 E. Feigenbaum, *The Handbook of Artificial Intelligence Volume I* (Reading, MA: Addison-Wesley) 1989

54 P. Winston, *Artificial Intelligence* (Reading, MA: Addison-Wesley) 1977

55 R. Drazovich and B. McCune “Radar with Sight and Knowledge” (Fairfax, VA: Journal of Electronic Defense) June 82

56 I. Levitan and L Kaczmarek, *The Neuron: Cell and Molecular Biology* (Oxford: Oxford University Press) 1997

57 D. H. Johnson. *Signal Encoding in the Firing Patterns of Auditory Neurons*, Invited talked, IMA Workshop on Audition, March, 1999

58 E. H. Shortliffe, MYCIN, A rule-based computer program for advising physicians regarding antimicrobial therapy selection (Artificial Intelligence in Medicine, 251, October 74), and *MYCIN: Computer-based Medical Consultations* (American Elsevier) 1976

59 A. Kozlov, *Efficient Inference in Bayesian Networks* Doctoral Dissertation (Palo Alto, CA: Stanford University) 1998

60 A. Dempster, “Upper and Lower Probabilities Induced by a Multivalued Mapping” *Annals of Mathematical Statistics* #28, 1967

61 L. Zadeh, "Outline of a new approach to the analysis of complex systems and decision processes" *IEE Transactions SMC-9* (NY: IEEE Press) 1973

62 J. Baltzersen, *Qunilan's Algorithms and the Rough Sets Approach Project Report 1995* (Trondheim, Norway: The Norwegian Institute of Technology) 1995

63 M. Gupta et al, *Approximate Reasoning in Expert Systems* (Amsterdam: North-Holland) 1985

64 R. Sutton and A. Barto *Reinforcement Learning* (Cambridge, MA: MIT Press) 1998

65 S. Kaufman *At Home in the Universe* (Santa Fe, NM: The Santa Fe Institute) 1992

66 R. Schoonderwoerd et al, "Ant-like agents for load balancing in telecommunications networks" *Autonomous Agents 97* (Marina Del Ray, CA: ACM) 1997

67 K. Clark and S.-A. Tarnlund Eds, *Logic Programming* (London: Academic Press) 1982

68 J. Minker, *Prolog Course Notes* (College Park, MD: U Maryland) 1983

69 T. Finin, "KQML -- A Language and Protocol for Knowledge and Information Exchange," *International Conference on Building and Sharing of Very Large-Scale Knowledge Bases*, Tokyo, December, 1993.

70 N. O. Bernsen, H. Dybkjaer, and L. Dbykjaer, *Designing Interactive Speech Systems* (London: Springer) 1998

71 www.worldlanguage.com/systran

72 K. Goodman and S. Nirenburg *The KBMT Project: a case study in Knowledge-Based Machine Translation* (San Mateo, CA: Morgan-Kaufman) 1991

73 R. Berwick, *The Acquisition of Syntactic Knowledge* (Cambridge, MA: The MIT Press) 1985

74 J. Allen, *Natural Language Processing* (Prentice-Hall) 1997

75 "IBM ViaVoice Online Companion Empowers Internet Applications With Speech Technology" (http://www.speechtechnology.com/news/04029901.shtml) 2 April 2000

76 R. Rodman, *Computer Speech Technology* (Boston: Artech House) 1999

77 E. De Mori *Computer Models of Speech Using Fuzzy Algorithms* (NY: Plenum Press) 1983

78 J. Allen, "Natural Language Understanding" Chapter XIX, *The Handbook of Artificial Intelligence Volume IV*

(Reading, MA: Addison-Wesley) 1989

79 H. Gish, GTE Natural Language Processing Systems, personal communication (Cambridge, MA: GTE BBN) 1999

80 Sixth Text REtrieval Conference (TREC-6)NIST Special Publication 500-240 (Gaithersburg, MD: National Institute of Standards and Technology) August 1998

81 SNePS (Internet: ftp.cs.buffalo.edu:/pub/sneps/) 1998

82 Koser, et al, "read.me" www.cs.kun.nl:(The Netherlands: University of Nijmegen) March 99

83 The XTAG Research Group, *A Lexicalized Tree Adjoining Grammar for English* Institute for Research in Cognitive Science (Philadelphia, PA: University of Pennsylvania) 1999

84 PC-KIMMO Version 1.0.8 for IBM PC, 18-Feb-92

85 WebL Programmers Guide (Compaq Computer Corp) 1998

86 A. Karmouch, "Guest Editorial: Mobile Software Agents for Telecommunications" *IEEE Communications Magazine* (NY: IEEE Press) July 1998

87 Esfandiari et al, "An Interface Agent for Network Supervision" *Intelligent Agents for Telecommunications Applications* (Amsterdam: IOS Press) 1998

88 Albayrak, "Introduction to Agent Oriented Technology for Telecommunications" *Intelligent Agents for Telecommunications Applications*, Edited by S. Albayrak (Amsterdam: IOS Press) 98

89 T. Finin, "KQML -- A Language and Protocol for Knowledge and Information Exchange," *International Conference on Building and Sharing of Very Large-Scale Knowledge Bases*, Tokyo, December, 1993.

90 van Aeken and Demazeau, "When Agents Talk" *Intelligent Agents for Telecommunications Applications*, Edited by S. Albayrak (Amsterdam: IOS Press) 98

91 K. M. Chao et al, "Sharing of Domain Knowledge" *Intelligent Agents for Telecommunications Applications*, Edited by S. Albayrak (Amsterdam: IOS Press) 98

92 V. Anh Pham and A. Karmouch, "Mobile Software Agents: An Overview," IEEE Communications Magazine (NY: IEEE Press) July 1998

93 Busuioc "Distributed Intelligent Agents – A Solution for the Management of Complex Services" *Intelligent Agents for Telecommunications Applications* (Amsterdam: IOS Press) 1998

94 J. Sztipanovitis and G. Karsai, "Self-adaptive software for signal processing" Comm. of the ACM, Oct 98

95 B. Abbott et al, "Model-based Approach for Software Synthesis" IEEE Software (NY: IEEE Press) May 93

96 J. Reilly, "Secure Intelligent Agent Extensions to the TMN Management Framework" *Intelligent Agents for Telecommunications Applications* (Amsterdam: IOS Press) 1998

97 Rouveirol et al, Specification of the Common Knowledge Representation Language (CKRL) of the MLToolbox, P2154/D2.2 (Marcoussis, FR: (Alcatel) MLT Consortium) 1993

98 SB-ONE (Internet: kobsa@inf-wiss.uni-konstanz.de), 1998

99 BACK (a KL-ONE derivative) back@cs.tu-berlin.de; ftp.cs.tu-berlin.de:/pub/doc/reports/tu-berlin.de/kit/Back52; Files are BACK_V52.intro and Back52.tar.Z, 1998

100 CLASSIC (A KL-One Derivative) (Internet: dlm@research.att.com) 1998

101 KRIS (a KL-ONE derivative) (Internet: baader@dfki.uni-kl.de) 1998

102 Luke, et al, "Ontology based Web Agents" Autonomous Agents 97 (Marina Del Ray, CA: ACM) 1997

103 MOTEL (Internet: hustadt@mpi-sb.mpg.de; mpi-sb.mpg.de:/pub/tools/motel.tar.Z [139.19.1.1]) 1998

104 RELFUN (Internet: boley@informatik.uni-kl.de) 1998

105 FORWARD (Internet: hinkelma@dfki.uni-kl.de) 1998

106 Teleos (Inernet: www.ai_faq_general_part5.html) 1998

107 URANUS (Internet: etlport.etl.go.jp:/pub/uranus/ftp/; Hideyuki Nakashima <nakashim@etl.go.jp>) 1998

108 BinNet Web Site www.binnet.com, Dec 99

109 COLAB/ TAXON (Internet: hanschke@dfki.uni-kl.de) 1998

110 T. Winograd, *Language as a Cognitive Process*, (Reading, MA: Addison Wesley) 1983

111 Kaplan and Bresnan "The Lexical-functional grammar: A formal system of grammatical representation" *The Mental Representation of Grammatical Relations*, (Cambridge, MA: MIT Press) 1982

112 COLAB/CONTAX (Internet: meyer@dfki.uni-kl.de) 1998

113 Williams and Cagan, Activity Analysis: "The Qualitative Analysis of Stationary Points for Optimal Reasoning", AAAI 94, AAAI, Menlo Park, CA, 1994.

114 Goto and Muraoka, "A Beat Tracking System for Acoustic Signals of Music" Multimedia 94 (San Francisco, CA: ACM) 1994

115 Muller et al Expert Systems in Telecommunications Management *Conference Record Globecom 93*, p. 883 (NY: IEEE Press) Nov 93

116 Noy and Hafner, "The State of the Art in Ontology Design", AI Magazine, (Menlo Park, CA: AAAI) Fall 97

117 Barbuceneau and Fox, "Integrating Communicative Action, Conversations and Decision Theory to Coordinate Agents" Proceedings of Autonomous Agents 97 (ACM 97)

118 T. Finin et al., "Mobile Agents Can Benefit from Standards Efforts on Interagent Communication," IEEE Communications Magazine (NY: IEEE Press) July 1998

119 R. Ready, "The Hearsay II Speech Understanding System" *The Handbook of Artificial Intelligence* (Los Altos, CA: William Kaufman Inc) 1982

120 R. Drazovich and B. McCune, "Radar with sight and knowledge" (NY: Defense Electronics) 1983

121 R. Calistri-Yeh, "Applying blackboard techniques to real-time signal processing and multimedia network management" IEA/AIE '94, pp 593-599 (ACM) 1994

122 K. Fukunaga, *Introduction to Statistical Pattern Recognition* (San Diego, CA: Academic Press) 1990

123 Schaeffer, *Theory of Evidence* (Palo Alto, CA: Tioga) 1982

124 C. Shannon, The Collected Works (NY: IEEE Press) 1990

125 Earl and Firby, "Combined Execution and Monitoring for Control of Autonomous Agents" Proceedings of Autonomous Agents 97, (Marina Del Ray, CA: Association for Computing Machinery)

126 A. Rao and M. Georgoff "BDI Agents: From Theory to Practice", Technical Note 56 (www.umbc.edu: Australian Artificial Intelligence Institute) 1995

127 D. Dowe, *The Mixture Modeling Home Page* (Internet: http:// www.csse.monash.edu.au/ ~dld/

mixture.modelling.page.html), 1998

128 Orwant, *Perl 5 Interactive Course* (Corte Madera, CA: Waite Group Press) 1998

129 Tcl/Tk web site (http://www.sco.com/Technology/tcl/Tcl.html)

130 E.L. Antworth, *PC Kimmo Reference Manual* (www.linguistictools.org: Summer Institute of Linguistics) 1990

131 E. L. Antworth, PC-KIMMO: a two-level processor for morphological analysis. Occasional Publications in Academic Computing No. 16. (Dallas, TX: Summer Institute of Linguistics ISBN 0-88312-639-7) 1990

132 www-cgi.cs.cmu.edu/afs/cs.cmu.edu

133 R. C. Schank *Conceptual Information Processing* (Amsterdam: North Holland) 1975

134 SNePS (Internet: ftp.cs.buffalo.edu:/pub/sneps/) 1998

135 A. Pisano, *MEMS 2003 and Beyond*, (www.darpa.mil/mto/mems: DARPA) 1999

136 G. Minden et al, *Rapidly Deployable Radio Networks API* (U Kansas, 1998)

137 Signal Processing WorkSystem (Foster City, CA: Alta Group of Cadence Design Systems) 1994

138 Object GEODE (Paris, FR: Verilog) 1998

139 J. Mitola III, Software Radio Architecture: A Mathematical Perspective IEEE JSAC (NY: IEEE Press) April 1999.

140 J. Minker, *Logic Programming Course Notes* (College Park, MD: University of Maryland) April 85

141 J. P. Bowen, Formal Methods Home Page (www.reading.ac.uk) 1999

142 M. Herlihy, and N. Shavit "A Simple Constructive Computability Theorem for Wait-Free Computation" STOC 94 (Quebec, Canada: ACM) May, 1994

143 M-P Gervais, and A. Diagne, "Enhancing Telecommunicaitons Service Engineering with Mobile Agent Technology and Formal Methods," IEEE Communications Magazine (NY: IEEE Press) July 1998

144 P. Winston, *Artificial Intelligence* (Cambridge MA: MIT Press) 1982

145 E. Davis, "The Naïve Physics Perplex", AAAI Magazine (Palo Alto, CA: The American Association for Artificial Intelligence), Vol 19, No. 4, Winter 98

146 J. Pearl, *Causality: Models, Reasoning, and Inference* (San Francisco: Morgan-Kaufmann) Mar 2000

147 International Telecommunications Union, *Recommendation X.900 Data Networks and Open System Communications Open Distributed Processing* (Geneva: ITU) 1997

148 A. Beguelin et al, *Users Guide to Parallel Virtual Machine* (Oak Ridge, TN: Oak Ridge National Laboratory) 1991

149 P. Cohen, "Neo: Learning Conceptual Knowledge by Sensorimotor Interaction with the Environment" Proceedings, Autonomous Agents 97 (ACM) 1997

150 J. Gunn, "A Low-Power DSP Core-Based Software Radio Architecture" IEEE JSAC (NY: IEEE Press) April 99.

151 T. Kanter *Adaptive Personal Mobile Communication*, Licentiate Thesis (Stockholm, Sweden: The Royal Institute of Technology (KTH)) Mar 2000

152 E. Kaplan (Editor), *Understanding GPS : Principles and Applications* (Boston: Artech House Telecommunications Library) 1996

153 Russian Design Bureaus, AIRCRAFT GUIDANCE AND CONTROL: PRINCIPLES OF AIRCRAFT INS, SINS AND GPS DESIGN (www.cwa.ru) Mar 2000

154 A. Wiesler and F Jondral, "Software Radio Structure for Second Generation Mobile Communication Systems", *Proceedings of the 48th International Vehicular Technology Conference* (Ottawa, Canada: IEEE Press) May 1998

155 A. Wiesler, R. Machauer, and F Jondral, "Comparison of GMSK and linear approximated GMSK for use in Software Radio", *Proceedings of ISSSTA '98* (Sun City, South Africa: IEEE Press) September 1998

156 A. Wiesler, H. Schober, R. Machauer, and F Jondral, "Software Radio Structure for UMTS and Second Generation Mobile Communication Systems" *Proceedings of the 50th International Vehicular Technology Conference VTC'99 Fall* (NY: IEEE Press) September 1999

157 A. Kulkarni et al, "Implementation of a Prototype Active Network" *OPENARCH '98, San Francisco, CA* (NY: IEEE Press) April 1998

158 A. Kulkarni and G. Minden "Composing Protocol Frameworks for Active Wireless Networks" IEEE

Communications Magazine (NY: IEEE Press) March, 2000.

159 C. Phillips, “Optimal Time-Critical Scheduling” STOC 97 (www.acm.org: ACM) 1997

160 K. Curie and A. Tate, “O-Plan: The Open Planning Architecture” Artificial Intelligence 52 (1) 49-86, 1991

161 J. Albus et al, *Final Report (Task Force on Intelligent Control)* (NY: IEEE Press) December, 1993

162 S. K. Das et al, “Decision making and plan management by intelligent agents: theory, implementation, and applications” *Proceedings of Autonomous Agents 97* (www.acm.org: ACM) 1997

163 L. Esmahi et al, “Mediating conflicts in a Virtual Market Place for Telecommunications Network Services” *Proceedings of the 5th Baiona Workshop on Emerging Technologies in Telecommunications* (Vigo, Spain: Universidade de Vigo) 1999

164 A.-I. Mouaddib “Progressive Negotiation for Time-Constrained Autonomous Agents” Autonomous Agents 97 (www.acm.org: ACM) 1997

165 www.tetramou.com

166 M. Nouri, *TETRA Standard Interfaces and Gateways* (www.tetramou.com: Marconi) mar 2000

167 Selic et al, Real-Time Object-Oriented Modeling (NY: John Wiley & Sons, 1994)

168 Y.-B. Ko and N. H. Vaidya, “Geocasting in Mobile Ad-hoc Networks: Location-Based Multicast Algorithms” 0-7695-025-0/99 (NY: IEEE Press) 1999

169 C. R. Lin and J.-S. Liu, “QoS Routing in Ad-hoc Wireless Networks” JSAC (NY: IEEE Press) Aug 99

170 J. Mikkonen, *Quality of Service in Radio Access Networks* (Tampere, Finland: Tampere University of Technology) May 1999.

171 P. Meche, *UWC-136 Self Evaluation* (Irving, TX: Nokia Mobile Phones) Aug 98

172 Analytica Users Guide © Lumina Decision Systems (Denver, CO: Decisioneering) 1998

173 Stallings, Handbook of Computer-Communications Standards, Volume1, The Open Systems Interconnection (OSI) Model and OSI-Related Standards (NY:Macmillan, 1987).

174 http://it.kth.se/~jmitola/CR1

175 D. W. Kahn, *Topology* (NY: Dover Publications) 1995

176 J. Schuster and R. Luebbers, *A Hybrid SBR/GTD Approach for Modeling UHF Radio Wave Propagation in Urban Environments* (State College, PA: The Pennsylvania State University) Feb 1997

177 A. Vander Vorst et al "From Electromagnetics to System Performance: A New Method for the Error-rate Prediction of Atmospheric Communication Links" IEEE JSAC (NY: IEEE Press) May 1997

178 I. P. Schkarofsky and S. B. Nickerson, "Computer modeling of multipath propagation: Review of ray-tracing techniques" Radio Science, (American Geophysical Union) 1982

179 E. Lutz et al, "The Land Mobile Satellite Communications Channel – Recording, Statistics, and Channel Model" *IEEE Transactions on Vehicular Technology* (NY: IEEE Press) May 1991

180 W. R. Braun and U. Deresch, "A Physical Mobile Radio Channel Model" *IEEE Transactions on Vehicular Technology* (NY: IEEE Press) May 1991

181 Prop*Sim* Multipath Radio Propagation Simulator Product Information (Oulu, Finland: Elektrobit Oy) 1993

182 www.jpython.org

183 RFCAD 2.3 (www.comm-data.com) 1999

184 WRAP Software Tool (Vaxjo, Sweden: Enator Communications AB) 1999

185 Bertonie et al, "UHF Propagation Prediction for Wireless Personal Communications", Proceedings of the IEEE (NY: IEEE Press, Sep 94)

186 Berlitz, (1989) Swedish Phrase Book and Dictionary, Princeton, NJ: Berlitz Publishing Company, Inc.

187 http://ilk.kub.nl/

188 S. Hakkarainen, *Dynamic Aspects and Semantic Enrichment in Schema Comparison* Ph.D. Dissertation 99-009 (Stockholm, Sweden: Stockholm University) 1999

189 Yfantis et al "On time alignment and metric algorithms for speech recognition" Proceedings, Conference on International Information Intelligence and Systems (NY: IEEE Press) 1999

190 J. Zavrel and W. Daelemans, *Recent Advances in Memory-based Part of Speech Tagging* (Tilburg, The Netherlands: Tilburg University) 1999

191 B. Busser et al, “Machine Learning of Word Pronunciation: The Case Against Abstraction” (www.kub.nl) 1999

192 J. Fodor, *Modularity of Mind* (Cambridge, MIT Press) 1983

193 R. Sun and T. Peterson, “Some experiments with a hybrid model for learning sequential decision making” Information Sciences 111 83-07 (Amsterdam, The Netherlands: Elsevier) 1998

194 J. Liberti and T. Rappaport, Smart Antennas (Wiley Interscience) 1999

195 D. Traum and P. Dillenbourg, “Miscommunication in Multi-modal Collaboration” *Proceedings of the AAAI workshop on Detecting, Preventing, and Repairing Human-Machine Miscommunication* (Palo Alto, CA: AAAI) 1996

Appendix A Software Radio Architecture Evolution

Foundations, Technology Tradeoffs, and Architecture Implications

(Invited Paper)
Joseph Mitola III
The MITRE Corporation, McLean, VA, USA, and
KTH, The Royal Institute of Technology, Stockholm, Sweden

***Abstract* – Software radio has emerged as a focus of both academic research and commercial development for future wireless systems. This paper briefly reviews the foundation concepts of the software radio. It then characterizes the tradeoffs among core software-radio technologies. Object-oriented analysis leads to the definition of the radio reference platform and the related layered object-oriented architecture supporting simultaneous hardware and software evolution. Research issues include layering, tunneling, virtual machines and intelligent agents.**

KEYWORDS: Software Radio, Digital Radio

I. FOUNDATIONS

An architecture is a framework in which a specified class of components is used to achieve a specified family of functions (e.g. communications services) within specified constraints, the design rules [1]. Industry organizations including the Software-Defined Radio (SDR) Forum are in the process of defining open-architectures for SDR [2]. The wireless functions of an SDR architecture are shown in Figure 1.

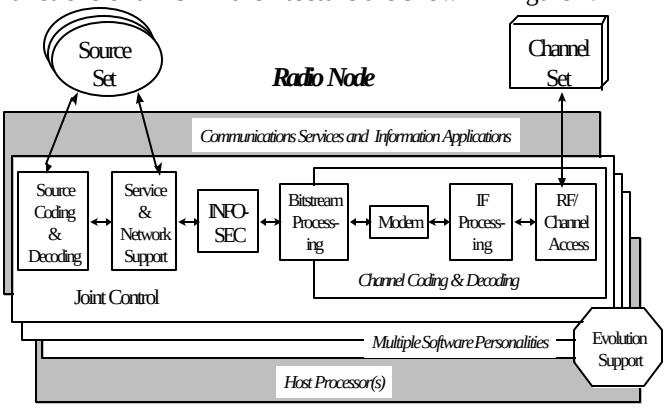


Figure 1. Functional Model of a Software Radio Node

A. Functions of the Software Radio

Technology advances are enhancing the physical-layer flexibility of wireless devices. Multiband antenna and radio frequency (RF) technology [3, 4], now enable access to more than one RF band at once. The ***Channel Set*** therefore includes multiple RF bands. Personal Communications System (PCS) base stations [5] and mobile military radios [6] can also use fiber and cable, also included in the channel set. Channel coding encompasses programmable ***RF/ Channel Access, IF Processing, and Modem.*** Multiband antennas and RF conversion comprise the RF/ Channel Access function. RF functions may include interference suppression [7]. ***IF Processing*** may include filtering [8]; further frequency translation; joint space-time equalization [9]; integration of space diversity, polarization or frequency diversity channels [10]; digital beamforming; and smart antennas [11]. ***Bitstream processing*** includes Forward Error Control (FEC) and soft-decision decoding. Although many applications do not require ***Information Security (INFOSEC),*** there are incentives for its use. For example, authentication reduces fraud, and stream enciphering ensures privacy. INFOSEC may be null for some applications. The ***source set*** may include voice, data, facsimile, video and multimedia. Some sources are physically remote from the radio node, e.g. connected via the Synchronous Digital Hierarchy (SDH), a Local Area Network (LAN) [12], or other network through ***Service & Network Support***.

Multimode radios [13] generate multiple air interface waveforms ("modes") using the ***modem***, the RF channel modulator-demodulator. Waveforms may be in different bands and may span multiple bands. Each combination of band and mode is one of ***multiple personalities***. Each personality combines RF band, channel set (e.g. control and traffic channels), air interface waveform, protocol, and related functions. In a software radio, all these functions are implemented using digital techniques in multithreaded multiprocessor software managed by a ***Joint Control*** function. Joint control assures system stability, error recovery, and isochronous streaming of voice and video. Joint Control may evolve towards autonomous selection of band, mode, and data format [14].

The functions of Figure 1 may be singleton (e.g. single band), multiple, or null. IF-Processing may be null, for example, in a direct conversion receiver [15]. In addition, dynamic compilation of software and real-time switching among personalities can allow a set of radio functions to be

integrated into a data-driven component such as a Field Programmable Gate Array (FPGA). The personality of an FPGA varies as a function of the processing requirements represented in a packet header [16]. A software radio can download new personalities [2]. These personalities may modify any aspect of the air interface. The resource demands for isochronous performance (e.g. bandwidth, memory, and processing capacity) of the resulting personality must not exceed those available. ***Evolution support*** is therefore necessary to define the waveform personalities, to download them (e.g. over the air) and to assure that each new personality is safe before being activated. To type-certify such a radio, one must guarantee that the properties specified by the regulatory bodies are preserved in spite of this high degree of flexibility.

B. Classes of Software-Defined Radio (SDR)

The software-radio parameter space of Figure 2 represents radio implementations as a function of digital access bandwidth and programmability. The horizontal axis characterizes programmability in terms of the ease of making changes. For example, the HF STR-2000, a commercial product of Standard Marine AB shown at point (A) used baseband Analog to Digital Conversion (ADC), with DSP in the TMS320C30 for high programmability. Commercial Off

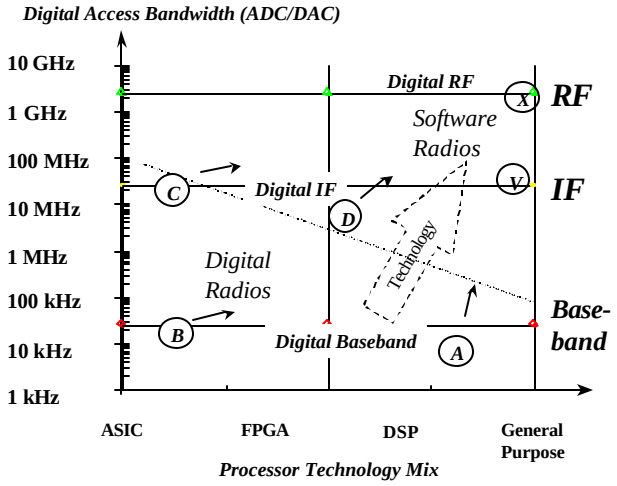


Figure 2. Software Radio Parameter Space

The Shelf (COTS) cellular telephone handsets fall near (B). Application Specific Integrated Circuits (ASICs) deliver processing capacity, shifting these designs toward the less programmable end of the axis. Digital cell site designs, (C), similarly, rely heavily on digital filter ASICs for frequency translation and filtering, even though they access the spectrum at IF. SPEAK*easy* II, (D), provides a GFLOP of programmable DSP, shifting this implementation to the right [17]. The Virtual Radio [18], (V), delivers a single channel radio using a general-purpose processor, DEC's (Compaq's) Alpha. Point (X) is the ideal software radio with digital RF and all functions programmed on a RISC processor. Although maximally flexible and thus of great research interest, such designs are currently economically impractical. This parameter-space quantitatively differentiates software radios ((V)-(X)) from Programmable Digital Radios (PDRs) ((A)-(D)).

PDRs may have more than one RF band and mode. The programmability is achieved via baseband DSPs; without digital IF, a PDR is not a software radio. PDR hardware modules, "slices," must be interchanged to change RF bands. Such a slice radio is a hardware-defined radio, not a an SDR. A multi-slice radio with all slices ***in*** the radio, selectable by software, on the other hand, is an SDR.

In an SDR transmitter, baseband signals are transformed into sampled channel waveforms via channel modem functions implemented in software that drives high performance DACs. These signals may be pre-emphasized or non-linearly pre-coded [19] by the IF processing software. A PDR is not an ideal software radio if any crucial aspect of the channel waveform is implemented using programmable hardware (such as a voltage-controlled oscillator) rather than using software (e.g. sin/cosine lookup table). The current generation of SDRs are evolving towards the ideal software radio as technology continues to advance. SDR architecture must accommodate this evolution, subject to the following technology tradeoffs.

II. TECHNOLOGY TRADEOFFS

Antenna architecture determines the number and bandwidth of RF channels. This constrains the number and bandwidth of ADCs. Some waveforms currently require dedicated ASICs (e.g. W-CDMA despreaders) instead of ADCs. Digital streams connect FPGAs, DSPs, and general-purpose processors yielding a multi-threaded, multi-tasking, multi-processing operating environment. This section characterizes the tradeoffs among these SDR platform technologies.

A. Antenna Tradeoffs

Flexible antennas, RF hardware, and IF processing is a major technology challenge for software radio. Optimum analog performance requires resonant narrowband antennas. As illustrated in Figure 3 (a), this results in multiple parallel antenna/ RF-conversion channels. In this example, a Personal Digital Assistant (PDA) accesses first generation (1G) cellular (AMPS), 2G digital cellular (PCS), or 3G waveforms in the 1G or 2G bands. For location-aware services, it has a GPS receiver. It also uses the corporate wireless LAN (WLAN).

One could fabricate such a PDA with 4 parallel RF-ASIC channels, a commodity GPS chip and a future low cost Bluetooth-class [20] wireless local interconnect.

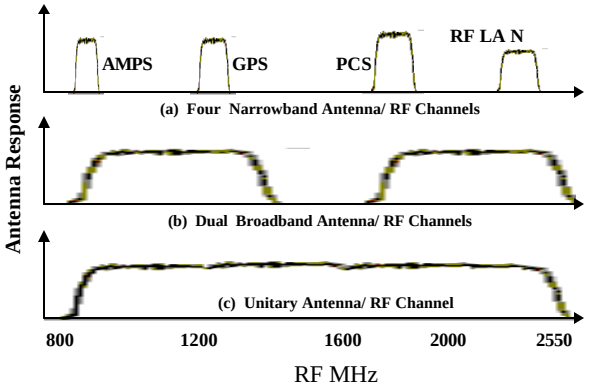


Figure 3. Antenna Tradeoffs

The broadband approach of Figure 3 (b) simplifies the antenna and RF to two parallel channels, reducing parts count. Figure 3 (c) shows a unitary wideband channel. The antenna response is not uniform across such a broad RF range. High performance in multiple RF bands drives one towards parallel narrowband channels. This can be an effective approach if cost is not at issue. Transmission efficiency and impedance matching is more challenging as bandwidth increases. Since antennas, RF conversion, IF processing and the ADC can account for over 60% of the manufacturing cost of an SDR, reducing the number of RF channels may be a significant design goal.

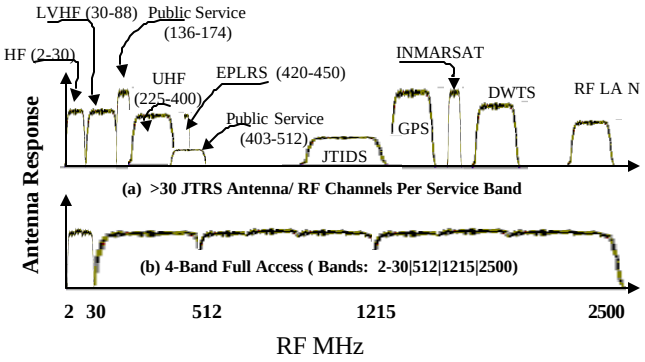


Figure 4. Four Software Radio Bands Span JTRS

Anticipating the Joint Tactical Radio System (JTRS) program of the US DoD [21], SPEAKeasy attempted to realize a unitary antenna [13]. The RF range extended from 2 MHz to 2 GHz, a ratio of 1000:1 or 3 decades. This requires a technology breakthrough, since the maximum relative-bandwidths of well-established designs are at most 10:1, one decade. Through in-depth technology tradeoffs, it was determined that at least 3 bands were needed. SPEAKeasy bands were: 1) 2-30 MHz; 2) 30-400 MHz; and 3) 0.4 to 2 GHz. Band 2 was implemented in SPEAKeasy I. Bands 1 and 2 were implemented in SPEAKeasy II. Currently affordable RF access is limited to less than one decade per channel. Therefore, a reference antenna configuration employs four high-performance bands (Figure 4 (b)).

B. RF and IF Processing Tradeoffs

The second tradeoff concerns RF and IF conversion. The transmitter may require both linear operation (e.g. for QAM waveforms) and non-linear operation (e.g. class-C amplifier for high power efficiency with FSK or PSK waveforms).

Single-channel receivers may non-linearly distort the waveform, e.g. in a direct-conversion architecture [15]. Multi-channel receivers (e.g. for cell sites), however, must provide linear response for the strongest and weakest subscriber signals ("near-far ratio," typically 90 dB). The RF and IF conversion linearity and dynamic range must match the ADC and Automatic Gain Control (AGC), and must support digital filtering and signal enhancement algorithms. The goal of this tradeoff is to balance the noise, spurious components, intermodulation products, and artifacts as illustrated in Figure 5. The noise floor is determined by the total bandwidth (e.g. in interference-limited bands below 400 MHz), or by the Low Noise Amplifier (LNA) e.g. in microwave bands. Spurious responses and Local Oscillator (LO) leakage sometimes can mask subscriber signals. LO leakage is problematic in homodyne receivers. A conservative design keeps the peak energy of all noise, spurs, and artifacts at about half of the Least Significant Bit (LSB) of the wideband ADC.

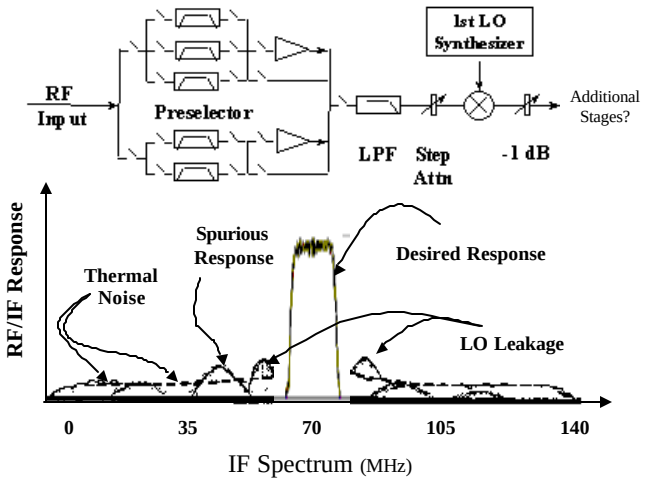


Figure 5. RF Tradeoffs Minimize Artifacts within Constraints

C. Interference Suppression

Antenna separation, frequency separation, programmable analog notch filters, and active cancellation suppress interference at the RF stage. A programmable interference suppression filter is illustrated in Figure 6. The filter is called a roofing filter because the interference sets the maximum linearly processable signal level ("roof"), while the dynamic range sets the minimum ("floor").

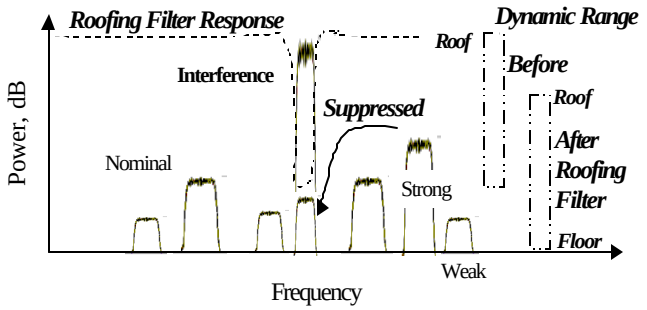


Figure 6. Workable Situation for Roofing Filter

Without the roofing filter, the roof of the dynamic range is so high that weak signals fall below the floor, resulting in dropped calls. With the filter, the roof is low so that the dynamic range reaches the noise floor. Roofing filters need low insertion loss (< 0.5 dB), programmable center frequency, and programmable bandwidth. Amplitude and phase ripple must be near zero to avoid distorting subscriber signals. If there are more than four interference signals, the roofing filters typically introduce excessive distortion.

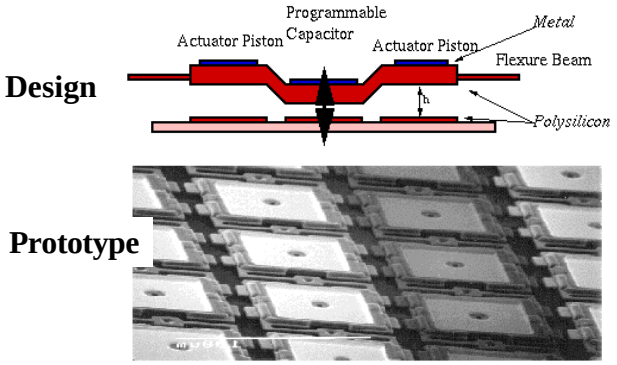
Active cancellation is the process of introducing a replica of the transmitted signal into the receiver so that it may be coherently subtracted from the input signal.

Wideband antennas and RF exacerbate interference. SDR algorithms can contribute to interference suppression. For example, a table of constraints may limit combinations of waveforms to a well-behaved subset. The constraint table specifies parameter limits on power, frequency, data rate, and number of simultaneous channels supported. The SDR monitors mutual constraint satisfaction to minimize self-generated interference.

D. RF MEMS

Most RF integrated circuits require off-chip resonators, inductors, and capacitors. Each discrete device increases the cost of production manufacturing, which is nearly a linear function of the number of parts (not cost per part). RF MEMS replaces these with on-chip 3D structures. For example, MIT developed a VLSI-compatible sealed cavity thin-film resonator (TFR) using piezoelectric films. TFRs exhibit a 1.36 GHz fundamental longitudinal resonance with a 3.5 dB insertion loss [22] and Q of 80,000 in 250 square microns. The device is six orders of magnitude smaller than discrete-component circuits. Wideband RF MEMS in GaAs and CMOS may be in production by 2001-2003.

MEMS RF switches are an electromechanical alternative to PIN diode switching circuits (needed to select RF path), substantially reducing size, weight, and power while improving performance. MEMS switches and tunable capacitors operate up to 40 GHz. In antenna interface units for airborne systems, they reduce size by 1000:1 and power by 10,000:1 while improving off isolation [23]. Continuing research in MEMS switch arrays targets a 1 Gbps data rate reconfigurable in 100 ns [24], a prototype of which is illustrated in Figure 7. Such components reduce the RF/IF device size, enabling multiband PDAs as an SDR-delivery platform.



Courtesy Professor Paul Franzon, North Carolina State University

Figure 7. High Performance MEMS Switch Fabric

E. Digital Architectures

An illustrative organization of DSP components for high performance SDR is shown in Figure 8. This abstraction may be used as a reference platform to the degree that it specifies functional groupings and interfaces but not design.

Many possible signal flows may be implemented on such a hardware suite. In an N-element array, the channel isolation filters extract channels for each of K subscribers on each of N elements. Algorithms in the DSP pool form beams. They also extract first-stage soft-decision parameters. Channels with low Carrier to Interference Ratio (CIR) are thus identified. Their bulk-delayed signals may be isolated for sequential interference cancellation, which also is performed in the DSP pool. This pool provides the processors for modulation and pre-distortion, including beamforming for transmission [25]. Switching functions employ the low-speed bus.

Matrix inversion for smart antennas substantially increases

the processing requirements, but yields improved performance. Consequently, many techniques have been investigated to reduce the computational burden of optimal algorithms, or to enhance the cancellation capability of simpler algorithms. A taxonomy of smart antenna techniques is provided in Figure 9.

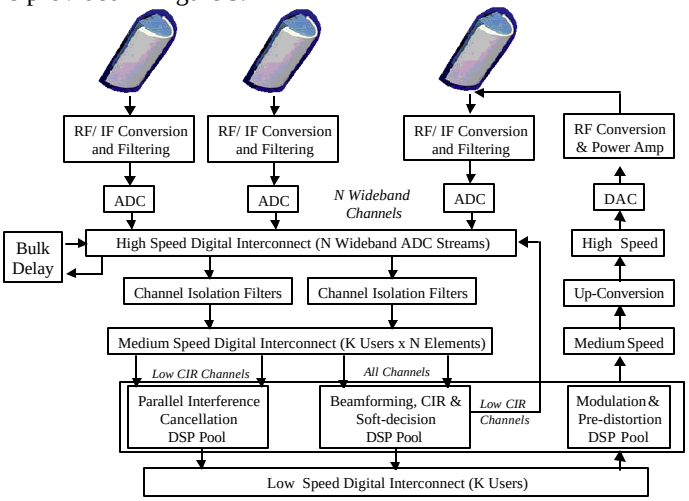


Figure 8. High Performance Digital Reference Platform

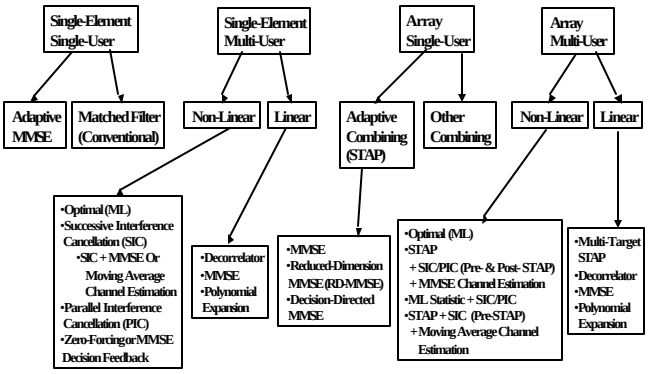


Figure 9. Smart Antenna Algorithms [26, 11]

Smart antennas will eventually dominate the digital hardware architecture of radio nodes, requiring 1 to 3 orders of magnitude more processing resources than a conventional node. Conventional nodes are supported by a smart-antenna architecture by nulling paths and components, but not conversely. The smart antenna architecture therefore provides a reference platform for SDR node evolution.

III. Architecture Analysis

Software functions may be organized into real-time objects. The hosting of these objects onto the complex SDR operating environment requires architecture analysis.

A. Architecture: Definition and Goals

Because of the open-ended nature of radio services and technology, architecture must support the evolution of new services, software, and hardware platforms. In addition, architectures should support enterprise-level component reuse. Industry-wide component reuse is called "plug-and-play." In an architecture that supports plug-and-play, the functional partitioning, component interfaces, and related design rules ensure that hardware and software modules from different suppliers work together when plugged into an existing system. Hardware modules require physical and logical interfaces that are compatible with the host hardware platform. Software modules require a comprehensive but simple interface to the software-operating environment. A module that offers its description to this environment may be integrated as a resource.

B. Layering and Virtual Machines

Protocol layering [12] is a well-established method of achieving some of the goals of radio architecture. For example, wireless Internet services are supported by the Wireless Application Protocol (WAP) [27]. WAP maps Internet applications to the limited data rate, connectivity, computation, and display limitations of cellular radio handsets. WAP therefore is an interface layer between applications and the radio platform. Access to the underlying platform is limited.

Java [28] provides increased access to the underlying computational engine of a handset. It provides more than WAP, but less than existing software (e.g. C). The Java Virtual Machine (JVM) defines a general purpose computing engine that hides the details of the computer's native Instruction Set Architecture (ISA). In addition, Java's input streams, output streams and related facilities hide the details of host operating systems, resulting in a platform-independent Internet applications language. Java, however, cannot access the underlying radio communications capabilities of a handset or PDA. It has no primitives to modulate a sine wave or tune a receiver. These are not yet elements of the JVM, therefore one might use Java's Native Interface (JNI) with radio-specific enhancements.

C. Object-Oriented Analysis

One approach to SDR architecture would be to extend Java with classes that access or implement such radio primitives. Ada, C, and C++ have been used to implement radio functions. Motorola's SPEAKeasy II Applications Programming Interface (API) [17] was the first public set of

radio primitives according to which one might define such an extension. This API was based on a set of software objects. Through object-oriented analysis, those objects evolved into the entity reference model of the SDR Forum [2]. This partitioning of functions into objects is similar to that of Figure 1. The SDR Forum's entities consist of Antenna, RF, Modem, Black[1] Processing, INFOSEC, Internetworking, System Control, and Human-Computer Interface.

Implementation of these entities requires more than WAP and Java. WAP is tailored to wireless but not suited to integrating heterogeneous radio applications like a modem object and an IF Filter object. Java interpreters may be computationally inefficient compared to C and assembler. One would like objects whose internal structure could be computationally efficient such as: C, an encapsulated FPGA, or part of an ASIC. Object request broker (ORB) technology provides the interface needed among software modules. The Common Object Request Broker Architecture (CORBA) and its associated Interface Definition Language (IDL) implement efficient interfaces among software objects [29]. The SDR Forum adopted CORBA as its middleware. With this approach, radio objects use facilities of a CORBA-based Core Framework (CF) to access radio facilities and computational resources. The CF includes framework control, a repository of software resources, a file manager, and a resource manager as illustrated in Figure 10.

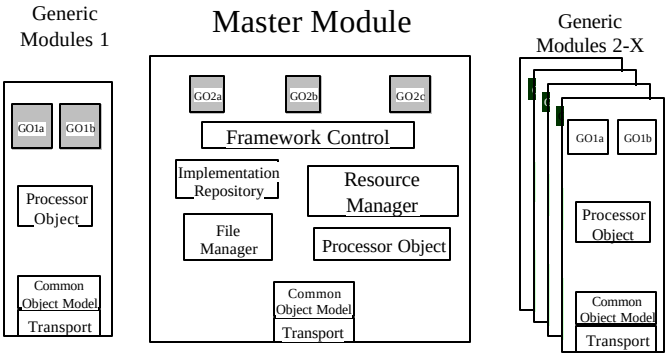


Figure 10. SDR Forum Core Framework [2]

Entities conform to the Common Object Model which links SDR entities across distributed processors via the Processor Object abstraction.

IV. Research Issues

Several research issues arise in using such architectures to support the evolution of the software radio. First, the computational stability of the radio objects is undefined. Second, a mechanism for characterizing the radio-related capabilities of the hardware platform is needed. Third, many applications need direct access to hardware facilities. Finally, the integration of multiband multimode services is left up to the user. Each of these research issues are addressed below.

A. Computational Stability

Mathematical analysis of software radio includes the modeling of radio architecture using point-set topology [1]. Radio objects may be modeled as maps among topological spaces as illustrated in Figure 11. The IF-Waveform space, for example, defines the interfaces between the dual-band antenna and the IF processing operations.

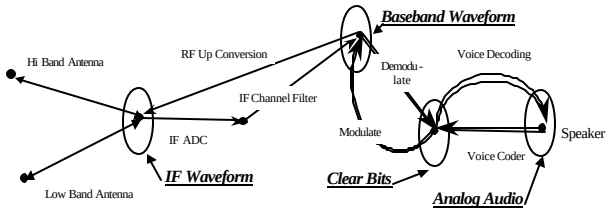


Figure 11. Topological Model of Dual Band Handset Streams

The integration of radio objects (e.g. using CORBA) is topologically equivalent to the composition of maps. If each map is a partial-recursive function, then the evolving radio is partial. Two radio objects that are well-behaved separately may cause the system to crash by attempting to use excessive processing or memory resources. Radio objects, unlike general purpose computing objects, are isochronous. Radio objects must run-to-completion within a fixed time-window. They therefore do not need Turing-computability. The bounded recursive subset of the total recursive functions is a sufficient subset for radio objects. This guarantees that the use of computational resources may be computed in advance. Loop bounds may be derived from the duration of the isochronous window. This can preclude unbounded iterative loops. Consequently, arbitrary combinations of such bounded objects are computationally stable. For example, a genetic algorithm may combine modem and protocol objects to autonomously evolve new radio protocols. Without computational stability, such a radio endangers a network. With assured computational stability, one obstacle is removed from the autonomous evolution of software radios. An immediate practical benefit is the simplification of type-certification by removing the need to exhaustively test all combinations of radio personalities [1]. An architecture that prescribes bounded recursion achieves these benefits.

[1] Black is military jargon for encrypted data.

B. Hardware Reference Platforms

With a variety of hardware implementations, it is difficult to determine whether a specific hardware configuration will support a specific software configuration. The radio reference platform abstracts the properties of the hardware environment that determine its capability to support classes of software modules. Table 1 identifies the hardware reference-platform parameters appropriate to a software radio. If these parameters are specified with precision and if the hardware conforms to the reference platform, then software developed for one member of the family will port readily to another member of the family.

Table 1. Software Radio Reference Platform Parameters

Critical Parameter	Properties
Number of Channels	Number of antenna/RF/IF channels
RF Access	Continuous coverage
Digital Bandwidth	Maximum ADC/DAC per channel
Dynamic Range	End to end (RF, ADC, processing)
Interconnect Bandwidth	For those buses, ports, backplanes, etc. that limit throughput
Timing Accuracy	Precision and stability of clock(s)
Frequency Performance	RF, IF, and Local Oscillator (LO) accuracy and stability
Processing Capacity (MIPS, MFLOPS)	Use standard benchmarks, (per processor class if appropriate)
Memory Capacity	RAM, ROM; mass storage
Hardware Acceleration	Parameterize encapsulations (despreader ASICS, FPGAs, etc.)
Operating Environment	OS, CORBA, APIs, with benchmarked throughput

Such a table defines a top-level reference platform. The smart antenna reference platform introduced above, suitably parameterized, is an example of a detailed reference platform.

C. Direct Access to Hardware Facilities

Process tunneling is the direct access to hardware facilities using a single, lightweight software wrapper. This bypasses intervening layers of middleware, operating system, etc. With this approach, the application invokes the hardware as if it were a software object. A tightly integrated, efficient software wrapper passes arguments and control signals to the hardware directly. This wrapper object accommodates the unique ISA and operating system requirements. The Virtual Radio's GUPPI interface [18] is an example of tunneling. Virginia Tech's Soft-Radio FPGA reconfiguration facility is another example [16].

Figure 12 shows how tunneling and virtual machines may be integrated with CORBA and radio applications objects. The radio infrastructure layer incorporates resource management, timing, frequency distribution, middleware [26], and tunneling. These facilities enable software to evolve in platform-independent computer languages (e.g. Java) or ORB-compliant software modules. At the same time, the tunneling wrappers permit efficient access to ASICs and FPGA personalities.

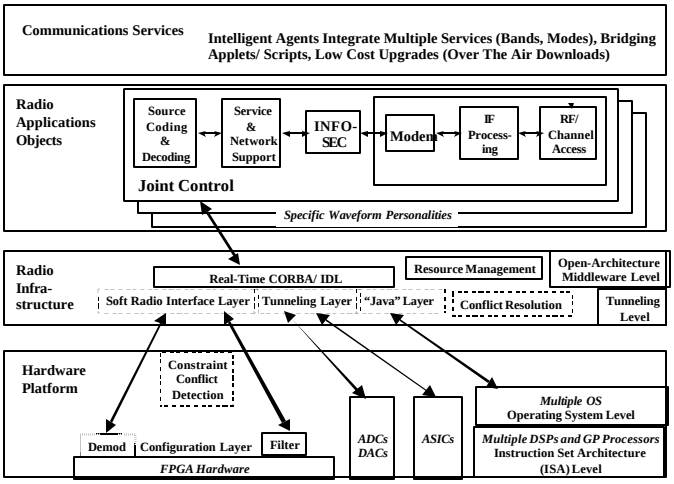


Figure 12. Layered Virtual Machine with Tunneling

Constraint detection, introduced earlier, is extended to detect conflicts in access to hardware. The radio applications layer embodies radio objects at whatever level of abstraction is convenient. Waveform personalities may be monolithic wholes accessed by a radio API from the application layer; or they may be partitioned into an FPGA-based reconfigurable Front-end with a CORBA-based back end; or they may be partitioned into SPEAKeasy II objects as suggested in the figure; or SDR Forum entities. The lowest layer conforms to some radio reference platform, such as the smart antenna platform. The layered virtual machine architecture therefore provides a flexible basis for the evolution of current PDRs towards the ideal software radio.

D. Service Integration

The deployment of 3G, the proliferation of wireless LANs, and the integration of GPS, video, thermal sensors, etc. into PDAs offer a bewildering array of alternatives to consumers. Enhanced autonomy has been identified as a means of integrating these alternatives into customized services. Autonomous agents may use a Radio Knowledge Representation Language as a basis for inferring user needs for wireless bands and modes from use-context [14]. PDAs with propagation-modeling tools may even be able to rent radio spectrum from each other [30].

V. CONCLUSION

Industry-standard SDR architecture will shape the degree to which waveform plug-and-play is realizable for the wireless marketplace. The SDR Forum has made significant progress in defining such an architecture. Research issues include better understanding of the computational properties of heterogeneous radio objects. Additional layering, tunneling, and virtual machines extend the current architectures to a basis for the graceful evolution of SDR hardware and software components. As such architectures allow complexity to increase rapidly, agent technology will be needed to integrate services in a context-sensitive way, further propelling this evolutionary process.

Disclaimer: This paper is not endorsed by The MITRE Corporation, the JTRS Joint Program Office, nor the US DoD

---

1 J. Mitola III, "Software Radio Architecture: A Mathematical Perspective" JSAC (NY: IEEE Press) April 99

2 SDR Forum Technical Report V2 (www.sdrforum.com) 99

3 P. Poggi, "Applications Of High Efficiency Techniques To The Design of RF Power Amplifier And Amplifier Control Circuits" MILCOM '95 (NY: IEEE Press) 1995

4 Bennett, D., "The ACTS FIRST Project And Its Approach To Software Radio Design" Proceedings of the 4th ACTS Mobile Communications Summit (EC) 99

5 Mouly and Pautet, *The GSM System for Mobile Communications* (Plaiseau, France: Mouley & Pautet) 1992

6 Smith, D. "A Perspective On Multi-Band, Multi-Mission Radios" MILCOM (NY: IEEE Press) 1995

7 U. Rhode et al, *Communications Receivers* (NY: McGraw-Hill) 1997

8 Zangi and Koilpillai, "Software Radio Issues in Cellular Base Stations", JSAC (NY: IEEE Press) 1998.

9 Martone, M., "Cumulant-based Adaptive Multichannel Filtering for Wireless Communications Systems … Using Antenna Arrays", IEEE TVT (NY: IEEE Press) May 98

10 Morgensen and Petersen, "Practical Considerations of Using Antenna Diversity in DECT" 0-7803-1927-3/94 (NY: IEEE Press, 1994)

11 J. C. Liberti, Jr., & T. S. Rappaport, *Smart Antennas for Wireless Communications* (Prentice Hall) 1999

12 Stallings, *Handbook of Computer-Communications Standards, Volume1, The Open Systems Interconnection (OSI) Model* (NY:Macmillan, 1987).

13 Upmal and Lackey, "SPEAKeasy, the Military Software Radio" IEEE Comms Magazine (NY: IEEE Press) '95

14 J.Mitola, *Cognitive Radio: Model-based Competence for Software Radios*, Licentiate Thesis, (Stockholm: KTH, The Royal Institute of Technology)August, 1999

15 Abidi, "Low Power RF IC's for Portable Communications," Proc. IEEE (NY:IEEE Press) Apr 95.

16 S. Srikanteswara et al, "A Soft Radio Architecture for Reconfigurable Platforms" IEEE Communications Magazine (NY: IEEE Press) Feb 2000

17 Cook, P., "An Architectural Overview of the Speakeasy System" IEEE JSAC, April 99

18 Bose et al, "Virtual Radio" IEEE JSAC on Software Radios, (NY: IEEE Press) 1998

19 Feher, K, Wireless Digital Communications (Prentice Hall, Upper Saddle River, NJ), 1995

20 www.bluetooth.com

21 Operational Requirements Document (ORD) For The Joint Tactical Radio (JTR) (Washington, DC: US DoD) 1997

22 "Piezo Resonators…" EE Times 27 Jan 97

23 A. Pisano, *MEMS Principal Investigator's Meeting* (Washington, DC: System Planning Corporation) Jan 99

24 P. Franzon, "Low-Power, High-Performance MEMS-based Switch Fabric" (www.ncsu.edu: North Carolina State University) 1999

25 J. Evans, et al, "The Rapidly Deployable Radio Network" IEEE JSAC (NY: IEEE Press) April 99

26 J. Mitola III, *Software Radio Architecture* (NY: Wiley Interscience) 2000

27 Wireless Applications Protocol (www.wapforum.org)

28 www.javaworld.com, http://java.sun.com, etc.

29 T. Mowbray and R. Zahavi, *The Essential CORBA* (NY: John Wiley and Sons) 1995

30 J. Mitola III, "Cognitive Radio for Flexible Mobile Multimedia Communications" Proceedings of the Mobile Multimedia Communications Workshop (NY: IEEE) Nov 99

Appendix B: Software Radio Architecture[1]

A Mathematical Perspective

Joseph Mitola III
The MITRE Corporation

Abstract

As the software radio makes its transition from research to practice, it becomes increasingly important for researchers, product developers and service providers to develop insights into the mathematical foundations of software radio architectures. This paper contributes to this goal by critically reviewing the fundamental concept of the software radio, using mathematical models to differentiate this rapidly emerging technology from similar technologies such as programmable digital radios. The key resources of the software radio include programmable processing capacity and dynamically-defined services. The bounded recursive functions, a subset of the total recursive functions, are established as the largest class of functions for software radios with desired resource use properties. Analysis of the software architecture of a typical implementation yields a layered distributed virtual machine reference model and a set of architecture criteria for the software radio. The discussion introduces the topological properties of software radio architecture that promote plug-and-play applications.

KEYWORDS: Software Radio, Digital Radio, topology, computability

Table of Contents

[1] Published in the IEEE Journal on Selected Areas in Communications, Special Issue on Software Radio, May 1999, for which the author was the lead technical editor

1. Introduction

It is well established that a communications system is the set of devices by which one employs a communications channel to convey information to a recipient. From a radio engineering perspective, the communications device must first encode the information from the source into some suitable electronic representation. The device must then transform this internal form into a waveform compatible with the radio frequency (RF) communications channel. The channel distorts the RF signal, adds noise, and creates distorted replicas of the signal. The process is (imperfectly) reversed in the receiver as illustrated in Figure 1.

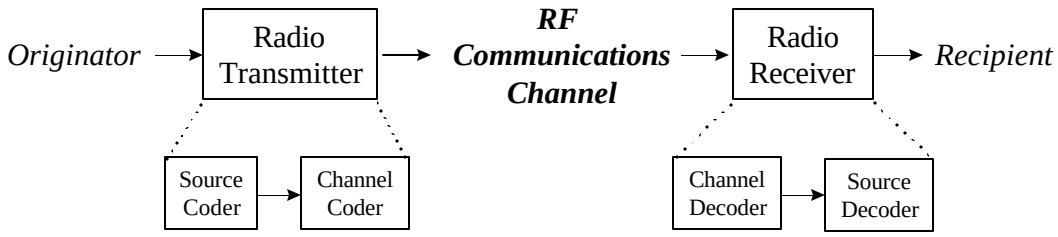


Figure 1. Traditional Model of a Radio Communications System -

In this venerable historical framework, control of the radio needs little attention, generally limited to power on/ off; audio volume control; a noise-riding receiver threshold ("squelch"); and a switch to manually select from among pre-defined RF channels. Several such transmitter(s) and receiver(s) located and working together comprise a radio "node." The multiband, multimode, multi-threaded, multi-personality capabilities of software radios require expansion of this model.

1.1 Essential Functions of the Software Radio

Technology advances have ushered in new radio capabilities that require an expansion of the essential communications functions of Figure 1. ***Multiband*** technology [1], first of all, accesses more than one RF band of communications channels at once. The RF channel of Figure 1, then, is generalized to the ***Channel Set*** of Figure 2. This set includes RF channels, but radio nodes like Personal Communications System (PCS) base stations [2] and portable military radios also interconnect to fiber and cable; therefore these are also included in the channel set. The channel encoder expands to ***RF/ Channel Access, IF Processing and Modem.*** Antennas and RF conversion that span multiple RF bands comprise the RF/ Channel Access function. ***IF Processing*** may include filtering, further frequency translation, space/time diversity processing, beamforming and related functions. ***Multimode radios*** [3] generate multiple air interface waveforms ("modes") defined principally in the ***modem***, the RF channel modulator-demodulator. These waveforms may be in different bands and may span multiple bands. The source and channel coders of Figure 1 therefore become the ***multiple personalities*** of Figure 2. A personality combines RF band, channel set (e.g. control and traffic channels), air interface waveform, and related functions.

Although many applications do not require ***Information Security (INFOSEC),*** there are incentives for its use. Authentication reduces fraud. Stream encipherment ensures privacy. Both help

assure data integrity. Transmission security (TRANSEC) hides the fact of a communications event (e.g. by spread spectrum techniques [4]). INFOSEC is therefore included in Figure 2, although the function may be null for many applications.

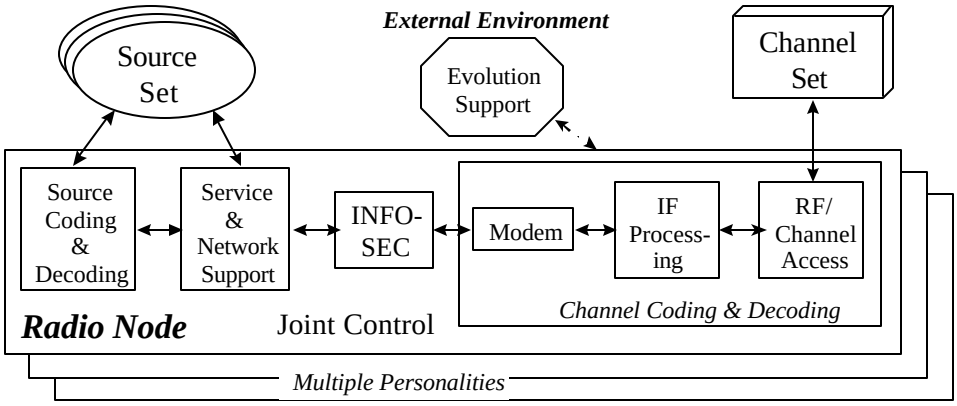


Figure 2. Functional Model of A Software Radio Communications System

In addition, the source coder / decoder pair of Figure 1 must now be expanded to include the data, facsimile, video and multimedia sources implicit in Figure 2. Some sources will be physically remote from the radio node, connected via the Synchronous Digital Hierarchy (SDH) [5], a Local Area Network (LAN) [6], etc., through ***Service & Network Support*** of Figure 2.

These functions may be implemented in multithreaded multiprocessor software orchestrated by a ***Joint Control*** function. Joint control assures system stability, error recovery, timely data flow, and isochronous streaming of voice and video. As radios become more advanced, Joint Control becomes more complex, evolving toward autonomous selection of band, mode, and data format. Any of the functions may be singleton (e.g. single band versus multiple bands) or null, further complicating joint control. Agile beamforming supports additional users and enhances quality of service (QoS) [7]. Beamforming today requires dedicated processors, but in the future, these algorithms may time-share a Digital Signal Processor (DSP) pool along with the Rake receiver [8] and other modem functions. Joint source and channel coding [9] also yields computationally intensive waveforms. Dynamic selection of band, mode, and diversity as a function of QoS [10] introduces large variations into demand, potentially causing conflicts for processing resources. Channel strapping, adaptive waveform selection and other forms of data rate agility [11] further complicate the statistical structure of the computational demand. In addition, processing resources are lost through equipment failures [12]. Joint control integrates fault modes, personalities and support functions on a limited resource of Applications-Specific Integrated Circuits (ASICs), Field Programmable Gate Arrays (FPGAs), DSPs and general-purpose computers to yield a reliable telecommunications object [13].

In a software radio, the user can upload almost arbitrary new air interface personalities. These may modify any aspect of the air interface, including whether the waveform is hopped, spread or otherwise constructed. The required resources (e.g. bandwidth, memory and processing capacity) must not exceed those available. Some mechanism for ***evolution support*** is therefore necessary to define the waveform personalities, to download them (e.g. over the air) and to assure that each new personality is safe before being activated. To type-certify such a radio, one must guar-

antee that the properties specified by the regulatory bodies will be preserved *in spite of this high degree of flexibility*. The need for such guarantees motivates the study of the mathematical properties of the software radio.

For example, one may model the statistical demand for computational resources versus processing capacity using queuing theory [14, 15, 16]. Real-time performance can be assured in a fixed architecture using this approach [17]. Plug-and-play, however, creates a variable architecture as modules are introduced into the environment and removed. This raises the complexity of the statistics, particularly in complex nodes such as a future cell site in which hundreds of users can invoke dozens of variable-bandwidth services in a pool of shared DSPs. Deeper understanding of the statistical properties of such environments is not doubt needed. And underlying such statistical analyses, there must be a predictable relationship of computational demand between the plug-and-play module and the environment. This calls for a theory of plug-and-play resource bounds for the software radio within which such predictable relationships will exist.

An obvious challenge is to define interface points for plug-and-play hardware and software modules. Industry organizations including the Modular Multifunction Information Transfer Systems (MMITS) forum are in the process of identifying such interface points using generalized Applications Programmers Interfaces (APIs) [18]. A less obvious challenge is to define architecture principles that assure that plug-and-play architectures will have the mathematical properties of controllability and predictability necessary for true plug-and-play services. These properties may be characterized in the following mathematical framework.

1.2 Towards A Mathematical Model of Plug-And-Play Architecture

The thesis of this paper is that the next-generation radio functions of Figure 2 have mathematical structure that defines a natural partitioning of the software radio into plug-and-play modules. The mathematical framework abstracts away the details of the radio functions, interfaces and implementations, identifying the computational and geometric structure that diverse software services and hardware platforms share.

In this framework, modules are modeled as mappings of signals among architecture interface points. These interface points are vertices in a topological space, a very general geometric space with a few set-theoretic axioms [19]. Mappings are represented as edges or arcs in this space. An edge is simply that which joins two vertices. An RF ASIC, for example, transforms the RF signal at the input vertex into the baseband signal represented by the output vertex. An arc, on the other hand, is a map that has properties in addition to its inputs and outputs, such as the power dissipated by the associated RF ASIC. This model may be used:

1. To identify top level plug-and-play interfaces;
2. To predict and control system performance;
3. To define a reference model that facilitates standards setting; and
4. To derive architecture principles for product evolution strategies.

Mathematical models of multiprocessor task specifications based on algebraic topology [20] have been used to prove important properties of shared memory multiprocessors. The model used in this paper is akin to such models. Rather than present the mathematical ideas formally and at once, they are introduced throughout the paper as needed. An illustrative model repre-

senting radio hardware and software as functional arcs joining the interface points in a topological space is shown in Figure 3.

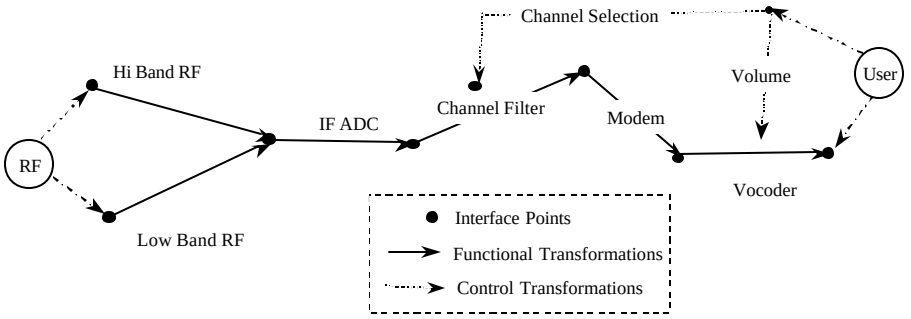


Figure 3. Topological Structure Of An Illustrative Radio

The abstraction in the figure shows a dual band receiver with an Analog to Digital Converter (ADC) at Intermediate Frequency (IF), a Channel Filter ASIC, and software Modem and Vocoder (voice coding) functions. Some arcs represent the movement of signals and data from one interface point to another via isochronous, real-time streams (e.g. the Hi Band RF hardware and Modem software of the figure). Others represent the exertion of control that translates a set of states (e.g. user choice of a channel) into a parameter that effects a stream (e.g. the Channel Filter). The properties of the arcs are also dimensions of the underlying (high dimensionality) topological space. So, for example, the node between the IF ADC and the Channel Filter would have the value "Hardware" in an "Implementation" dimension while the interface between Modem and Vocoder would have the value "Software". Other dimensions may be continuous, such as the range of settings of a control parameter. In addition, there may be infinite dimensional subspaces, e.g. to represent random components of analog signals. The dimensions, then, are chosen to represent the relevant features of the architecture.

1.3 Architecture Goals

A successful plug-and-play modular architecture entails at least the following:

1. **Compatibility:** The structure of plug-and-play modules must be compatible with that of the software radio environment - arcs must have nodes to plug into.
2. **Controllability:** Such modules must be controllable under module composition.
3. **Predictability:** Module composition must preserve radio service-defining properties of the system and when control is exerted, it must not have unintended consequences.

An architecture is a set of components used to achieve specified functions within specified constraints and design rules [21]. Compatibility, controllability and predictability are the properties of plug-and-play architecture studied in this paper. A mathematical framework for a plug-and-play architecture then should inductively establish desired properties of a given set of components that implement specified functions (services) within specified design rules. Due to the open-ended nature of radio services and technology, such a mathematical framework must be extensible both to new functions and to new implementation platforms. This paper begins the development of this framework with a top-down review of the properties of the software radio. A bottom-up analysis of the computational properties of software radio components then sets the

stage for the derivation of the layered virtual machine reference model. Architecture principles that support plug-and-play modularity are developed in the process.

2. Defining the Software Radio

This section defines the software radio from several perspectives.

2.1 Programmable Digital Radios

There are many ways to impart more than one personality to a radio. The Programmable Digital Radio (PDR) is a term applied to those radios that use a hardware-intensive mix of hardware and software techniques to access more than one RF band with a choice of air interface modes [22]. A PDR's programmability may be achieved using baseband Digital Signal Processing (DSP). However, hardware modules or "slices" typically must be swapped in order for the radio to change RF band and air interface mode.

The Software-Defined Radio (SDR) was defined by BellSouth to call for an evolution towards greater programmability of wireless infrastructure [23]. This evolution includes programmable multiband multimode radios implemented using the PDR approach. An IS-95/AMPS handset, for example, may employ a Code Division Multiple Access (CDMA) chip-set for IS-95; an Analog Mobile Phone System (AMPS) chip set; a dual mode analog RF Integrated Circuit (RFIC) chip-set [24]; a DSP chip [25] for filtering, voice coding, and other computationally intensive tasks; and a microcontroller for user interface and system control. Many of the functions are programmable, but the CDMA modem, for example, is defined in an ASIC and may not be changed in the field. Over time, functions initially implemented in hardware (e.g. the baseband modem) will migrate to software. Conversely, functions initially defined in software (e.g. in a simulation) may migrate to hardware (e.g. an ASIC). The mathematical framework must therefore capture the architecture implications of SDR migration between hardware and software.

2.2 The Software Radio

The software radio, on the other hand, imparts multiple personalities to a radio in a specific way [26]. A software radio defines all aspects of the air interface including RF channel access and waveform synthesis (not just selection) in software[2]. In the software radio, then, wideband analog to digital and digital to analog converters (ADCs [27] and DACs) transform each RF service band among digital and analog forms at IF as in Figure 4. The single resulting digitized stream of bandwidth Ws accommodates all subscriber channels, each of which has bandwidth Wc where Wc << Ws. The mathematical framework must differentiate alternative ADC and DAC designs, with the attendant differences in waveform programmability.

[2] Software here means algorithms stored in random access memory that run on general purpose computers and which may in principle be downloaded. Firmware, by this definition, is software to the degree that is can be changed during radio operations; if it cannot be changed, it is part of the hardware personality of the radio, not part of the software.

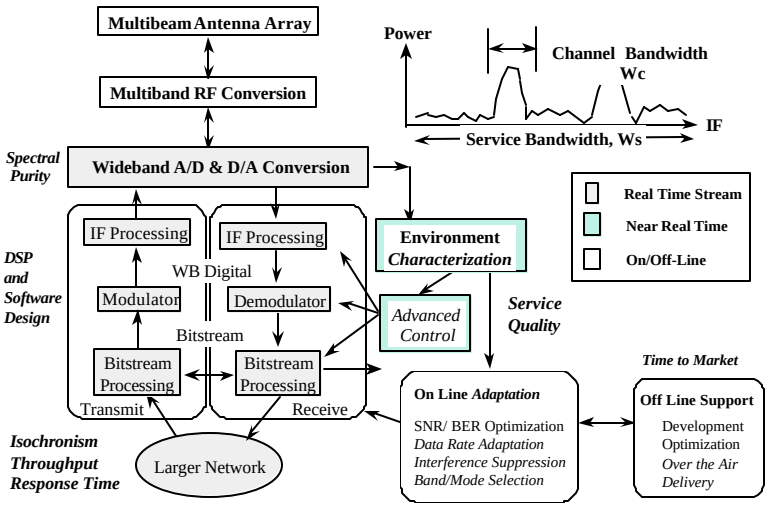


Figure 4 Key Software Radio Functions and Components

In the software radio illustrated in Figure 4, IF ADC and DAC channels are processed isochronously through digital hardware, interfaces and DSP software. IF processing may include filtering to isolate subscriber channels [28]; digital formation of nulls and beams [29]; integration of space diversity [30], polarization or frequency diversity channels [31]; and other means of acquiring a high quality waveform. There may be multiple IF frequencies (in the heterodyne receiver) or the IF frequency may be zero (in the homodyne receiver [32]). IF Processing may be null, for example, in a direct conversion receiver [33]. The mathematical framework, then, must represent plug-and-play implications of such variations including optional "nulling" of functions.

Digital downconversion is the process of using the frequency domain periodicity of sampled bandpass waveforms to translate the waveform to baseband without analog heterodyning [34]. As preselection filters with the necessary performance are reduced in size and improve in performance (e.g. through superconducting RF filters [35]); and as local oscillator leakage is reduced, digital downconversion becomes more feasible. In a software radio transmitter, baseband signals are transformed into sampled channel waveforms via channel modem functions which may be implemented in software using high performance DACs and dynamically reconfigurable FPGAs. The resulting output signals may be pre-emphasized or non-linearly pre-coded [36] by the IF processor. In some implementations, modem functions, IF processing and RF Channel Access may be amalgamated into a single component such as a direct conversion receiver RFIC [33]. In addition, dynamic compilation of software or real-time switching among FPGA personalities can allow these discrete functions to be integrated into a single component [37]. The mathematical framework, then, has to represent both discrete and integrated implementations, which can have significant impact on cost, time-to-market and plug-and-play characteristics.

2.3 A Software-Equivalent Model of Hardware Modules

The ADC and DAC define the point at which functions are potentially software-defined. We may call this point the *digital access point*. This point may occur anywhere in the architecture. Relatively inflexible analog and digital hardware such as ASICs may be necessary, e.g. for direct conversion of the channel waveform to a bitstream in a handset. Although an ASIC may have a

few control parameters, the basic personality of optimized ASICs cannot be changed in the field (e.g. from one air interface standard to another). To change the air interface personality, one must replace the ASIC; switch among different ASICS; or switch among modes of one ASIC. In the tradeoffs among, size, weight, power, cost and flexibility, fixed-function ASICs generally consume less power, take less space, weigh less and cost less per device than the programmable DSP equivalent. Therefore, the increased flexibility of the software radio comes at a price. One way of representing these differences mathematically is to attribute equivalent computational capacity to the non-programmable devices as illustrated in Figure 5.

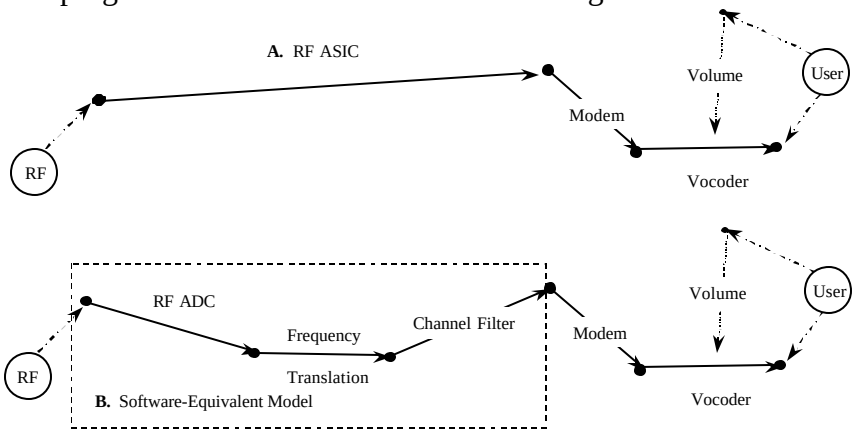


Figure 5 Topology of An RF ASIC (A) And Its Digital Equivalent (B)

In the topological model of the hypothetical radio (top), an RF ASIC (A) transforms the analog signal from the antenna directly to input for the baseband modem. The software-equivalent model (B, bottom) postulates an RF ADC of a bandwidth exactly corresponding to the bandpass characteristic of the RF ASIC. The frequency translation behavior of RF ASIC A is modeled as the Frequency Translation software process in B. The filtering behavior of the RF ASIC is represented in the Channel Filter software of model B. The sequence RF ADC, Frequency Translation and Channel Filter arcs constitute the software-equivalent model of this ASIC. This software is defined in terms of a canonical processor which is specified in conjunction with a given architecture model. Equivalent computational capability of RFIC (A) is the aggregate of the computational demand of the postulated software processes (B). Since analog hardware has variable performance over time and over different devices, the exact filter model of a given RF ASIC will change over time and will differ from other devices. Thus, each device is represented in a *set* of models. The single arc A with its associated input and output nodes characterizes RF ASIC A at a given point in time. The uncertainty principle further assures that one cannot measure the properties of a device without changing them. Each model therefore includes parameter tolerances to account for such uncertainties. These tolerances are represented by “open balls” [38] in the model parameter subspace of the topological space (Figure 6). One may also aggregate multiple device models into a parametric model by taking set unions. The set of all such arcs and open balls comprises the complete topology of the device.

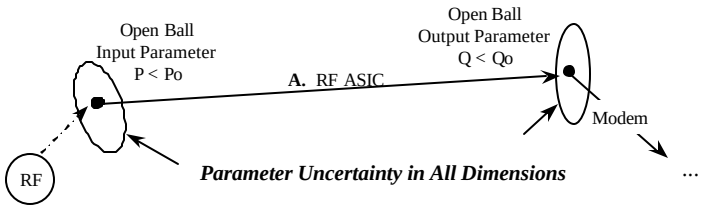


Figure 6 Parameter Uncertainty Represented by "Open Balls"

Each device may also have parameter sets in other dimensions that characterize or predict such diverse properties as internal timing, power dissipation, computational capacity with respect to a given benchmark, etc. Although the sequel focuses on computational properties, this set theoretic framework is general, capable of representing local properties of a device including interface details, size, weight, and power. Since topological spaces are not necessarily linear, aggregate properties of the radio system in the higher order subspaces may have complex, non-linear relationships to the properties of the components.

2.4 Quantifying Degrees of Programmability

The *degree of programmability* of an implementation is fundamental to software radio architecture. Since contemporary radios are made with a mix of processor types, one must characterize this mix from a software radio perspective. Consider the highest level arc-node model of a radio. Each arc may be hierarchically divided into its primitive constituent arcs. Those that may be redefined in software in the field may be labeled. The number of labeled primitive arcs divided by the total number of arcs is a measure of the programmability of the device. Since an ASIC's programmability is limited to the modification of a few parameters, most of its gate-level arcs will not be labeled.

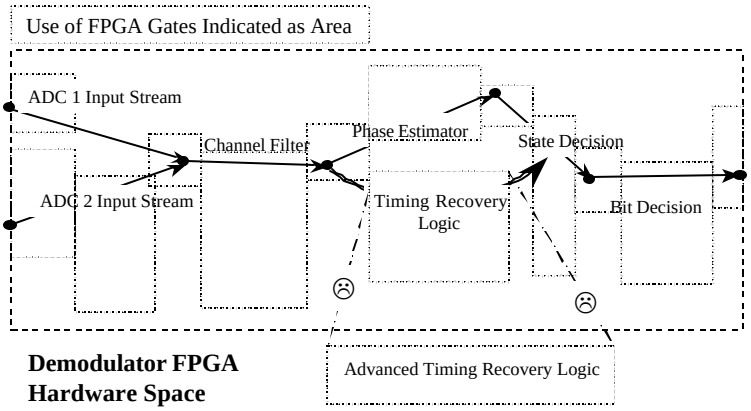


Figure 7 Hardware Topology Indicates Incompatibility of Download

FPGAs are in principle completely programmable. In practice, they are more programmable than ASICs, but subject to gate and interconnect constraints. Programmable radios have been based almost entirely on reconfigurable FPGAs [39]. Intuitively, however, the field-programmability of an FPGA is more constrained than that of a DSP chip, in part because of the possibility of running out of usable gates on the FPGA. The topological model of each type of device allows one to characterize ease of programmability. In the FPGA topology of Figure 7, two dimensions of Euclidean space (e.g. states and interconnect) represent the commitment of

logic to specific hardware. Suppose an advanced timing recovery algorithm, comprising, say, 10% of the FPGA area is to be downloaded to the radio. As shown, it is incompatible with the gate use of the existing timing recovery logic. It may be possible to redefine the entire personality of the FPGA to accommodate the new logic. In this case, the download bandwidth increases from the 10% needed for the increment to 100% of the personality, a 900% increase in the size of the download. It is also possible that a moderately populated (70%) FPGA will be unable to accommodate the 10% download because of hardware constraints such as the required placement of I/O buffers, lack of state registers where needed, etc.

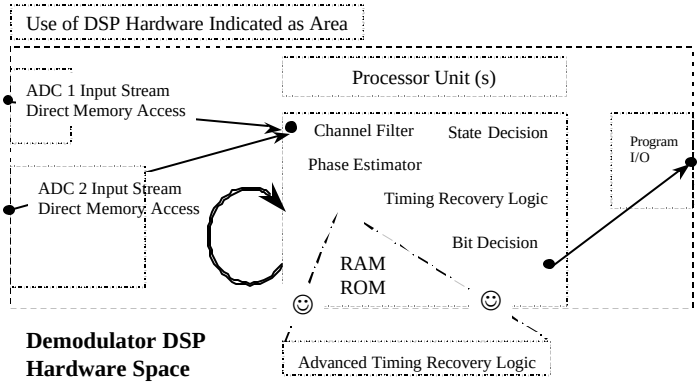


Figure 8 DSP Hardware Commitment Subspace Topology

A similar topological model of a DSP chip with multiple Direct Memory Access (DMA) channels is shown in Figure 8. The additional 10% of DSP code associated with the advanced timing recovery logic (now implemented in software) is accommodated provided the Random Access Memory (RAM) or electronically programmable Read Only Memory (ROM) has available space. In this case, the memory map allocates logic to RAM/ ROM hardware. DSPs may appear easier to program than FPGAs because RAM allocation can be accomplished by a compiler while allocation of logic to gates and interconnect in an FPGA generally requires an experienced designer. New waveforms also seem to outgrow the gates on an FPGA more easily than they outgrow the program memory of a DSP subsystem. On the other hand, the topology of timing constraints of DSP software may be more constraining than the timing of an equivalent FPGA. This is in part because logic in an FPGA can run at the system clock rate, while the speed of DSP code may be one to three orders of magnitude less than the system clock. The interplay of ASIC hardware allocation and DSP task timing is reflected in such topological models.

Complex or Reduced Instruction Set Computers (CISC/RISC) provide less hardware acceleration than DSP chips. To quantify the degree of flexibility of DSP, CISC and RISC processors, one again defines an appropriate topological space. Let {ISA} be the space of single-instruction register-state transformations of a processor with a given Instruction Set Architecture (ISA). From an arbitrary initial state, a DSP has many more edges connecting reachable data states than a CISC processor that in turn has more arcs than a RISC processor. So, from the perspective of {ISA} topology, the RISC processor is the simplest and thus in some sense the most general programmable hardware platform, CISC is more complex and DSP the most complex of the fixed

ISA machines. Performance of high quality code underscores these differences. DSP code that employs zero-overhead loops with full register stacks and processing elements yields higher throughput than CISC code of a processor with the same system clock, memory and I/O delays. FPGAs, on the other hand, effectively have a variable ISA that makes them even more computationally efficient than DSPs, CISC and RISC machines, with an attendant loss of generality.

These relationships define a phase space for the software radio. Physicists use a phase space to represent states of a substance (e.g. solid, liquid and gas) as a function of parameters such as temperature and pressure. The software radio phase space of Figure 9 represents the states of radio implementations as a function of the digital access point and the degree of programmability. The vertical axis represents the bandwidth at the digital access point. The horizontal axis represents degree of flexibility, the fraction of functionality that may be changed "in the field" using plug-and-play software. To place a system in the phase space, one examines the ADCs and DACs, placing the digital access point where the functionality is fully programmable. One then examines the way in which the software radio functions of Figure 2 are hosted on the physical devices. One places the radio system on the horizontal axis according to the degree of programmability of the device technology, attributing equivalent "software" processing capacity to digital ASICs and FPGAs where necessary.

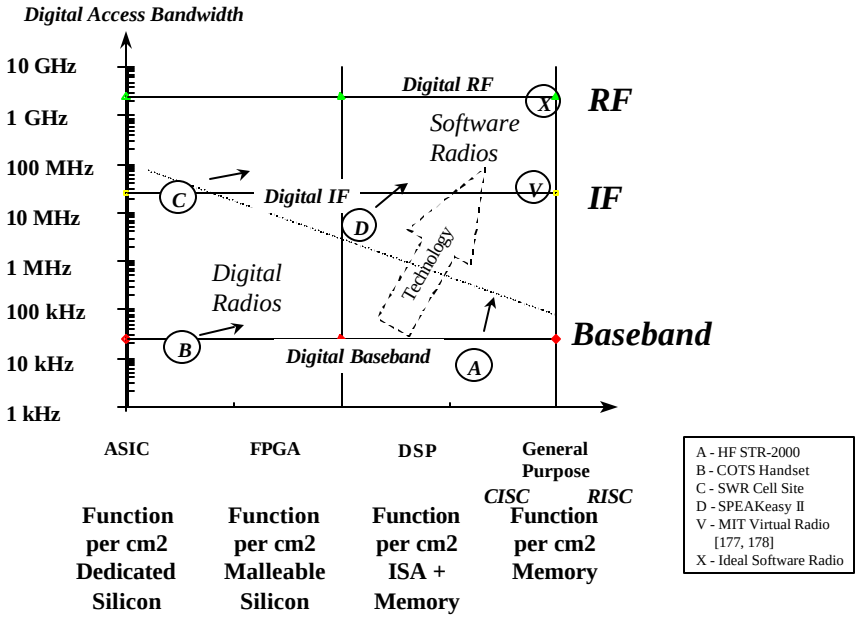


Figure 9 Software Radio Phase Space

Figure 9 places five types of radios in the phase space. The HF STR-2000, a commercial product of Standard Marine AB (A) employs baseband DSP using the TMS320C30. Most Commercial Off The Shelf (COTS) cellular telephone handsets fall near point B. ASICs provide much of the equivalent processing capacity, shifting these designs toward the less programmable end of the axis. Current software radio cell site designs, point C, similarly, rely heavily on digital filter ASICs for frequency translation and filtering, but they access the spectrum at IF. SPEAK*easy* II, point D in the phase space, provides considerable programmable DSP, shifting this implementa-

tion to the right. A product of research at the Massachusetts Institute of Technology, the Virtual Radio [40], delivers a single channel radio using a general-purpose processor, DEC's Alpha. This is the most general-purpose computing platform reported in the literature. Point X represents the Ideal Software Radio with the digital access point at RF and all functions programmed in general purpose RISC processors. Although maximally flexible and thus of research interest, such designs tend to be economically impractical. This phase-space quantitatively differentiates software radios in the upper right quadrant from the PDRs elsewhere in the figure.

A PDR is not a software radio if any crucial aspect of the channel waveform is implemented using programmable hardware (such as a voltage-controlled oscillator) rather than using software (e.g. sin/cosine lookup table). A Joint Tactical Information Dissemination System (JTIDS) radio [41] implementation with a 3 MHz IF from which hop frequencies are generated using a 250 MHz programmable local oscillator is ***not*** a software radio for that waveform. On the other hand, a 250 MHz digitized IF that could set every hop frequency in software would be a software radio. The radio may meet the software radio criterion for a large class of narrowband waveforms, while failing for wideband waveforms. A topological model of such a JTIDS radio would set the Implementation dimension of each arc required for the 250 MHz hopping as Hardware or Software. If the Implementation dimension of all such arcs is Software, then the implementation is a software radio. Such a model therefore unambiguously defines the specific degree to which a given implementation is reprogrammable. Such a model also identifies the components that must be changed in order to migrate from hardware to software and conversely.

3. Top Level Component Topology

Radio components that map stimuli to responses, conditions to decisions, etc. are represented as arcs in the topological model. An arc may be a collection of other arcs, defining a natural encapsulation hierarchy for the radio system. At the top level of the hierarchy, the radio node considered as a black box, maps air interface and user (or wire line) stimuli to appropriate responses. The functional components of Figure 2 define a second level of partitioning with attributes outlined in Table 1.

Table 1. Attributes of Top Level Software Radio Functional Components

Functional Component	Attributes	Remarks
Source Coding & Decoding	Audio, video, fax and data interfaces	Ubiquitous standard algorithms (e.g. ITU[42], ETSI[43])
Service & Network Support	Multiplexing; setup and control; data services; internetworking	Wireline and Internet standards including mobility [44]
Information Security*	Transmission security, authentication, non-repudiation, privacy, data integrity	May be null, but is increasingly essential in wireless applications [45]
Channel Coding & Decoding: Modem*	Baseband modem, timing recovery, equalization, channel waveforms, predistortion, black data processing, etc.	INFOSEC, modem, and IF interfaces are not standardized
IF Processing*	Beamforming, diversity combining, characterization of all IF channels	Innovative channel decoding for signal and QoS enhancement
RF Access	Antenna, diversity, RF conversion	IF interfaces are not standardized

Channel Set(s)	Simultaneity, multiband propagation, wireline interoperability	Automatically employ multiple channels or modes for managed QoS
Multiple Personalities*	Multiband, multimode, agile services, interoperable with legacy modes	Multiple *simultaneous* personalities may cause considerable RFI
Evolution Support*	Define & manage personalities	Local or network support
Joint Control* (over ***Channel Set***)	Joint source/channel coding, dynamic QoS vs. load control, processing resource management	Integrates user and network interfaces; multi-user; multiband; and multimode capabilities

** Interfaces to these functions have historically been internal to the radio, not plug-and-play*

The stream-oriented functional components are source coding and decoding; service and network support; information security; modem; INFOSEC; IF processing; and RF access. Each may be represented topologically by a pair of arcs between the domain data interfaces. A topological model of a specific radio consists of those arcs that correspond to the components as implemented. An example of a dual-mode handset signal-stream topology is provided in Figure 10. The dual band antenna and IF ADC are key aspects of this notional device which are readily apparent in the topological model.

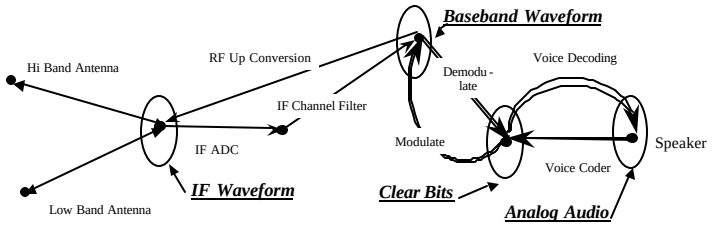


Figure 10 Topological Model Dual Band Handset Signal Streams

The top-level functional components of Table 1 share the data interfaces of Figure 11. Analog (audio and video) waveforms comprise the interface between Source Coding & Decoding functions and the Source Set, for example. Source bits define the interface between Source Coding & Decoding and Services & Network Support. In some cases, (e.g. in a handset) Services & Network Support may be null. If INFOSEC is null, protected bits are clear bits. In addition, some interfaces may be null from an implementation perspective. An RFIC, for example, may subsume the IF Waveform interface, exhibiting only Baseband and RF Waveforms in the highest level topology. Interface topologies are summarized in Table 2.

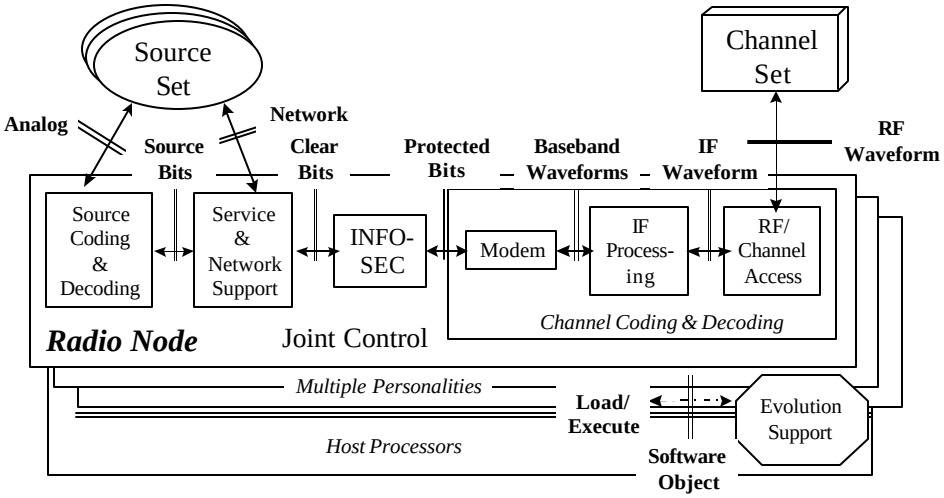


Figure 11. Minimum Interfaces Define Topological Properties

Table 2. Top Level Interface Topologies

Interface	Key Characteristics	Topological Properties
Analog Stream	Audio, video, facsimile streams	Infinite dimensional; Filtering constraints comprise open ball(s)
Source Bitstream	Coded bitstreams and packets. ADCs define a finite window into a quantized, discrete-time sampled waveform.	Finite dimensional; frame and data structure defines subspaces. Finite precision defines a Dynamic Range Subspace[3] for the ADC
Clear Bitstream	Framed, Multiplexed, Forward Error Controlled (FEC) bitstreams and packets	Finite dimensional; FEC subspaces have rich algebraic properties
Protected Bitstream	Random challenge, authentication responses; public key; enciphered bitstreams and packets	Finite dimensional; randomized streams; complex message passing for downloads; If null, interface reverts to clear bits
Baseband Waveform	Discrete time synchronous quantized sample streams (one per carrier)	Digital waveform properties determine fidelity of analytic representation
IF Waveform	Composite, digitally pre-emphasized waveform ready for up-conversion	Analog IF has infinite dimensional topology; Digital IF may have baseband product topology
RF Waveform	Power level, shape, adjacent channel interference, etc. are controlled	Analog RF has infinite dimensional topology; Includes spatial and temporal dimensions
Network Interface	Packaged bitstreams may require ATM, SS7, or ISO protocol stack processing	Synchronous Digital Hierarchy (SDH), Signaling System 7 (SS7) subspaces
Joint Control	Control interfaces to all hardware and software; initialization; fault-recovery	(Not illustrated in the figure) Parameter spaces; non-linear logic subspaces
Software Objects	Download from evolution support systems	Represents binaries, applets; includes self-descriptive language subspaces
Load/Execute	Software object encapsulation	Download topologies are highly nonlinear

[3] A Nyquist-Dynamic Range Subspace has been sampled so as to meet the Nyquist criteria for bandwidth recovery of the sampled signal and has been quantized with sufficient bits of sufficient accuracy to represent the two-tone spurious-free dynamic range of the application.

The analog stream interface, for example, may be represented in a topological space which includes not just the interface signal itself, x(t), but other meta-level characteristics: implementation, impedance, connector type, carrier frequency, bandwidth, etc. In addition, as shown in Figure 12, the topological representation structures the meta-level aspects of the interface. The complete definition of the interface spaces of a generic ratio is tantamount to defining a knowledge representation language (KRL) [46] for radio (RKRL), a significant undertaking that is beyond the scope of this introduction. Important properties of plug-and-play interfaces may be defined without a complete RKRL.

Domain:
Class: Analog-Stream
Implementation: Hardware
Signal-Interface: Coax-DC-Coupled
Impedance: 50 Ohm
Connector: BNC
Interface-Pin: 2-7
Carrier Frequency: Baseband
3dB Bandwidth: 350 kHz
Signals:
Predetected-Signal
Band-Limited-Signal
Interface-Signal: x(t)
Control Parameters:
Gain: 0dB to 20 dB
Gain Control: AGC

Figure 12 Illustrative Meta-level Topological Space for Analog Stream Interface

Such an interface defines a plug-and-play module if it is effectively separable from the rest of the system. Effective separability implies at least the following topological properties:

(a) Input and output subspaces are externally accessible in the host system.
(b) Composition of the functional transform of the module over the defined interface yields a well-defined function that results in the intended service.
(c) Control subspaces are compatible (not necessarily identical) with the host system
(d) Performance (e.g. spectral purity, data formats, throughput, response time, etc.) under function composition is within specified bounds

The separability of modules at plug-and-play interface points is illustrated in Figure 13. Plug-and-play modules may be defined by top level functional components such as the modem. Or, they may be defined at some arbitrary point in the hierarchy such as at the vocoder. The statistical, computational and geometric properties of the module shape system behavior. The goal is that the system as a whole behaves as desired if the plug-and-play modules and the host system each have the prescribed testable local properties. The system-level properties of interest in this paper center on reliable service delivery using bounded computational resources with guarantees against wait-induced fault conditions. Necessary interface properties include well defined be-

havior; isochronism of voice and video streams; and acceptable timing of control interfaces. While not an exhaustive list of desirable properties, these are essential for type-certification.

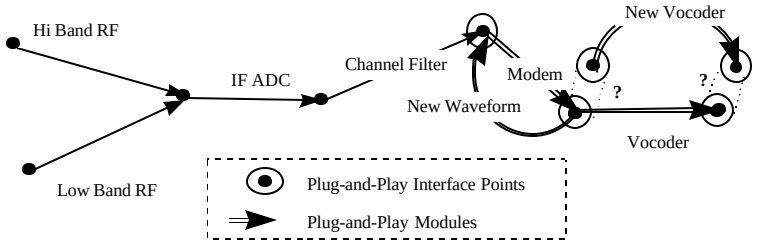


Figure 13. Topology of Plug-And-Play Interfaces

4. Computational Properties of Functional Components

A fundamental aspect of software radio architecture is the set of conditions under which plugging-in a module results in the intended system behavior. Topologists have studied the composability of functions defined over topological spaces. Homeomorphisms (topology-preserving maps) may be composed to yield other homeomorphisms provided the domain of one function and the range of the other are topologically compatible [38]. From a topological perspective, the domain of software consists of those inputs over which its results are defined and its range consists of the corresponding results (including side effects and returned values). Under what conditions can well-behaved software be composed with other well-behaved software to yield a well-behaved system? This very general question may be undecideable. But software radios are engineering systems with timing constraints that allow us to prescribe constraints on the topological structure of the software that establishes conditions under which composition of software modules is well behaved. The concept is that the bounded recursive functions are the largest set of functions for which predictably finite resource consumption may be guaranteed. In addition, the composition of bounded recursive functions is also bounded recursive, so we have our proof that defines conditions under which plug-and-play modules will not use excessive resources. Furthermore, there is no practical constraint on the radio functions that can be computed with the bounded recursive functions. This analysis addresses a crucial aspect of type certification: the ability to maintain throughput across changes of plug-and-play software modules.

4.1 Models of Computation

Earlier in this paper, the notion of software-equivalent hardware capability was introduced. To each ASIC and FPGA may be attributed the computational capability of a canonical processor which may in turn be modeled as a (collection of) von Neuman machine(s). Each procedure encoded in the software on a von Neuman processor may be modeled in terms of computability and concrete complexity as a Random Access Machine (RAM)[47]. A RAM consists of a state machine, input and output arrays; internal registers; the capability to load and store data via direct and indirect addresses; and the capability to increment memory values. The RAM model provides an intuitive but mathematically precise description of a single von Neuman processor. The RAM model has also been proven equivalent to the Recursive Function and Turing machine models of computing. Important theoretical properties include the ability to simulate in polyno-

mial time capabilities that require exponential time on Turing machines [48]. Let {RAM} represent the RAM model of the Instruction Set Architecture {ISA} of an arbitrary processor. Theorems outlined below use the Recursive Function and the RAM models to establish tight upper bounds on processing resources, a necessary condition for predictable throughput.

4.2 Primitive Recursive Functions

The recursive function model of computing consists of a set of functions from **N,** the natural numbers, onto **N** with closure properties that define classes of functions. The ***primitive recursive functions*** consist of the following functions:

1. The ***zero*** function, z: **N**->**N**: z(x)=0, which yields the constant zero for any input,
2. The ***successor*** function, s: **N**->**N**: s(x) = x+1, which associates a successor to each natural number,
3. The ***projection*** functions, U^n_m: $\mathbf{N_n}$->**N**: $U^n_m(x_1, x_2 \ldots x_n)=x_m$ by which arguments may be selected, with closure under:
4. ***Composition:*** for $g_1, g_2 \ldots g_m$: $\mathbf{N^n}$->**N** and h: $\mathbf{N^m}$->**N**, $f(\mathbf{x}_n) = h(g_1(\mathbf{x}_n), g_2(\mathbf{x}_n) \ldots g_m(\mathbf{x}_n))$, where $(\mathbf{x}_n)$ is brief notation for $(x_1, x_2 \ldots x_n)$, equivalently $f = h^*(\mathbf{g_m})$. That is, the set is closed under composition of functions. One can compose functions in a natural way that corresponds to function calls and/or selection of data inputs in a random access machine.
5. ***Primitive recursion:*** for g: $\mathbf{N^n}$->**N** , **h**:N^{n+2}->N, and finite y>0,
   $f(\mathbf{x}_n,y) =$ $g(\mathbf{x}_n)$ if y = 0, or
   $\mathbf{h}(\mathbf{x}_n, (y-1), f(\mathbf{x}_n, (y-1))$ for y>0.
   That is, the set of functions is closed under primitive recursion.

For y=1, and $\mathbf{h}(\mathbf{x}_n, 0, f(\mathbf{x}_n, 0)) = h(\mathbf{x}_n)$, f is simply the conditional execution of g or h based on the value of y [i.e. since $f(\mathbf{x}_n,0) = g(\mathbf{x}_n)$ but $f(\mathbf{x}_n,1) = \mathbf{h}(\mathbf{x}_n, 0, g(\mathbf{x}_n)) = h(\mathbf{x}_n)$]. This recursive pattern is equivalent to the if-then-else programming construct in terms of effective computability. Since f appears in its own definition, it requires a pushdown stack of successive arguments. But since initially y > 0 and y decreases at each invocation, the procedure will terminate and a stack which is as large as |y| will not overflow. It is assumed that the stack allocator in a specific {ISA} keeps the finite stack from overflowing, provided an upper bound for |y| is known in advance.

The ***primitive recursive functions*** are the smallest set of such functions closed under composition and primitive recursion. They include addition, subtraction[4], multiplication and others that do not require iterative search. Sequential logic; transversal and recursive filters; bit manipulation; and data packing are common primitive-recursive functions of software radios.

4.3 Total Recursive Functions

Iterative loops are not primitive recursive, but they are essential to radio software. The simplest model of iteration is bounded minimalization.

[4] Unconstrained subtraction and division are not primitive recursive, but appropriately constrained subsets are primitive recursive.

6. ***Bounded Minimalization:*** Let g: $\mathbf{N}^{n+1}$->**N** be primitive recursive; then f: $\mathbf{N}^{n+1}$->**N:** f (**x**, y) = $\mu z\{z<y\}[g(\mathbf{x}, z)=0]$ is also primitive recursive.

Read μz as "f (**x**, y) is *the least* z **less than** ***y*** for which g(**x**, z)=0". The class of functions closed under composition, primitive recursion and bounded minimalization are ***total.*** That is, they are defined for all **N.** The FORTRAN **do** loop exemplifies *bounded minimalization*. Ackerman's function [49] is defined everywhere but uses resources faster than any known function due to the structure of its calling sequences. It is not considered primitive recursive although it is a total function. Yet for given parameters of Ackerman's function which are known in advance, one can determine whether the function will exceed an allocated upper bound by using a suitable step-counting function. Any primitive recursive function admits a primitive-recursive step-counting function. Let the ***bounded primitive recursive*** functions [5] be the primitive recursive functions with the associated step-counting functions and finite resource bounds that are specified in advance. In software engineering terms, this model of computing insists that when a module is tested, its resource use in isolation (e.g. MIPS) be tested for bounding parameter sets. This is just good software-development practice for real-time systems. Integer division, exponentiation, generation of primes, finite logic predicates and finite iteration are examples of bounded primitive functions. These functions have properties attractive for software radio implementations as expressed in the following theorems:

Theorem 1 (Primitive Bounded Resources): Any bounded primitive recursive function is equivalent to a RAM program which terminates in a finite number of steps which can be bounded tightly from above, given bounds y_i, of the minimalizations from which the composite functions are created.

Outline of the Proof[6]: Associate a sequence of RAM instructions with each bounded primitive function and closure construct. Each corresponding RAM instruction sequence is primitive recursive and bounded. These RAM instruction sequences are like in-line subroutines. Each such sequence has a small finite number of steps (the bound). Compute the maximum concrete complexity of the recursive function structure from the {ISA} and a suitably chosen step-counting function [50]. Let the number of instructions from {RAM} be the step-counting function. The resources used by any bounded recursive function with a given set of calling parameters can be computed from the parameters and the step-counting function.

Lemma 1 (Primitive Bounded Execution Time): The dedicated-processor RAM-equivalent of any bounded primitive recursive function executes in an amount of time that may be tightly bounded. One determines the maximum execution time of each class of RAM instructions on a specific ISA. The execution time bound is the product of the number of steps from Theorem 1 times the maximum execution time for that class of instruction. The time used by the function tested in isolation on a specified dedicated machine is thus an alternative step-counting function.

---

[5] The class of total functions includes closure under restricted minimalization (unbounded minimalization defined everywhere). This class of functions is ***too big*** for software radios because its resource use cannot be guaranteed to be less than a bound specified in advance.

[6] Since these theorems are applications of well known results from the theory of computing, only the outline of the proof is given.

From a software engineering perspective, this lemma states that the bounds allocated may be specified in terms of execution time on a dedicated processor.

Relevance: Software radio includes isochronous streams for which there is a fixed timing window during which specific functions must be accomplished in order for services to be delivered properly and/or for the system to remain stable. Unconstrained modules may use resources that are within reason on test cases, but that "blow up" in other conditions. This approach is qualitatively different from much conventional practice. All primitive recursive functions can be guaranteed to either (a) complete within a specified window or (b) be ***easily computed in advance*** to be incapable of meeting that timing window (Lemma 1). Such predictable timing properties provide the foundation necessary to allocate computational resources to plug-and-play modules. If the resources available in the host are less than necessary, the plug-and-play module is not activated. Queuing theory [14] and operational analysis [51] translate such reliable service times measured in isolation into statistical bounds on isochronous performance in the service environment where other tasks are sharing the same processor(s).

4.4 Bounding The Partial Recursive Functions

Unfortunately, bounded primitive recursive functions do not express all of the programming constructs needed in software radios. Notably absent are the **while** and **until** loops. These loops search for a condition under which to terminate. This condition may never occur, so these RAM programs may loop forever. Hardware and software processes that wait or search for a condition that may not occur are computationally equivalent to while or until loops. Current research in wait-free computation considers the related problem of defining instruction sequences ("protocols") which assure that any process that itself is not in such a fault condition will terminate in a finite number of steps [59]. The source code of software radios like SPEAK*easy* I and II reveals the relatively widespread use of wait-prone constructs, e.g. waiting for sufficient signal strength to initiate receiver processing. The process-dispatch loop of the kernel operating system may "consume" infinite resources to deliver resources to applications software. Other infinite loops are computationally equivalent to the ***partial*** recursive functions. The computational structure is called (unbounded) minimalization:

> ***7. Minimalization:*** for g: $\mathbf{N}^{n+1}$->$\mathbf{N}$,
> $f(\mathbf{x}_n,y) = \mu y[\ g(\mathbf{x}_n,y) = 0]$,
> that is y is "the least y for which . $g(\mathbf{x}_n,y)$ is zero".

The search operator μ will continue to increment y and test **g** without bound. The partial recursive functions are the smallest set of functions closed under minimalization. Unlike bounded minimalization which consumes finite resources, the extent of which are computable in advance, unbounded minimalization consumes resources until g is satisfied. For many situations, g may never be satisfied. For example, a carrier detection loop that is waiting for a signal that will never be transmitted due to an impossible RF value will never terminate. It will keep using resources until someone steps in to force the release of the resources. The partial recursive functions have been proven equivalent to the Turing computable functions, the Post productions, and to the RAM model [52]. They compute essentially anything that we know how to compute.

Theorem 2 (Unbounded loops): Software radio functions implemented with while and/or until loops or their equivalents (e.g. implemented using go-to programming styles) cannot be guaranteed to use bounded computational resources.

Outline of Proof: Construct a while loop that simulates minimalization:

While(NOT(g($\mathbf{x}_n$, y)=0), increment y); Return y;

One may show the equivalence of until, while and for loops to unbounded minimalization. They are thus in not total and hence may be undefined. In these cases, the equivalent RAM procedure will consume unbounded computational resources or will cause an execution fault.

The instruction set {RAM} of a programmable processor always includes instructions by which such unbounded loops may be constructed. There are, however, practical ways which radio engineers and real-time programmers have devised for assuring system stability in spite of the regular use of such constructs. One may assign time limits to each search condition. A time-out yields an error state that deals with the excessive time to execute. Such programming techniques transform unbounded minimalization to bounded minimalization. An unbounded function may cause the radio to crash, while the bounded equivalent will cause the radio to enter an error-recovery state due to exceeding its time limit. Such limits must include the statistical effects of competition for resources with functions that have higher priority for system resources.

Radio applications are distinct from general purpose computing applications in that there are a rich set of timing constraints imposed on the creation, emission, reception, processing, conversion, network interoperation and user interface of radio signals. These timing constraints are often represented in message sequence charts and state machines. Timing constraints apply to call setup and control; delivery of isochronous voice, video and multimedia streams; packet networking; and related control systems. This is a heavily time-constrained applications domain. The constraints imply timing and hence resource bounds that can be allocated to individual hardware and software components in such a way as to constrain the logic (programming and hardware) to tightly bounded computational constructs without the loss of functionality. The assignment of such bounds supports a theory of stable software radios as follows.

Theorem 3 (Local Feasibility of Bounded Partial Recursion): Single-threaded single processor software radio functions may be synthesized in RAM instruction sequences equivalent to a resource-constrained subset of the partial recursive functions.

Outline of Construction (Local Bounded Partial Recursion): This proof builds on theorems 1 and 2. Unconstrained **While** and **Until** loops and their "go-to" equivalents which cannot be guaranteed to terminate are precluded. Each such construct from a conventional implementation is allocated a maximum number of iterations (or equivalently, a maximum processor time). These limits are coded into **bounded-while** or **bounded-until** loops as indivisible operations (in the same way one that a semaphore is an indivisible operation). Hardware that waits until a condition is met is similarly bounded, redesigned to generate a fault-interrupt if the condition is not met after a specified time interval: watchdog timers provide an example. Hardware and software "read" operations that wait for a response from a port (which may wait forever) are precluded. Poll and bounded-wait interrupt handshakes are substituted in which the system will continue

(with the read task suspended if the port has no data available). "Read" is also allowed in which the reading software task waits a specified time to be interrupted and then returns an error condition. Since there are RAM instruction sequences that implement these time-constrained search/wait loops, they are guaranteed to complete (either successfully or in a time-out state) in an amount of time that can be tightly bounded. This constrains the structure of the software (away from unbounded minimalization and hence from partial recursion) to bounded partial recursion. Since this construction applies to software within a given Von Neuman processor, the proof establishes the *local* feasibility of bounded partial recursion.

Theorem 4 (Totality): The bounded partial recursive functions are total [53].
Outline of Proof: Total functions are those that are defined for all **N.** Each of the primitive recursive functions is defined for all **N**. The closure under composition, primitive recursion and bounded minimalization is defined for all **N**. Bounded minimalization is defined for all **N** as follows. Associate with each range of arguments to a minimalization a step-counting function with a limit y_{max}. If the search terminates in less than $|y_{max}|$ steps, the loop yields the result of the minimalization. If not, then the result is FAIL for all $N > |y_{max}|$. Bounded minimalization is therefore total. The set of functions closed under such operations is therefore total.

Lemma 4 (Bounded Stability): The RAM instruction sequences that are equivalent to the bounded recursive functions are total. (Proof: If not, then the RAM model is not equivalent to the recursive function model, which would be a contradiction.)

Relevance: A software radio has to deliver services predictably. When combinations of inputs and states occur for which responses have not been defined, the system's behavior is unpredictable, hence it will produce undesired results and may ultimately go into an unrecoverable "crash" state. Software radios - in particular those with downloaded applications modules - have many modes and control parameters which result in a combinatorial explosion of control states which makes it difficult to test every combination of such states. However, if the software radio is constructed of modules that are each guaranteed to be total, then the state combinations will also be total by induction. Thus, some specific system behavior (non-crash) will be defined under all conditions. The *results* of such stable computations may or may not be as intended (i.e. "correct"), but if some specific state is defined, behavior will be ***repeatable***, so unintended results may be readily diagnosed and corrected. Crash states due to undefined unstable behavior, on the other hand, often require heroic labor-intensive debugging efforts. Radio systems fielded with software that is not total appear unreliable to the user.

Theorem 5 (Global Bounded Recursion): Multithreaded multiprocessor software-radio functions implemented using RAM sequences that are equivalent to the bounded recursive functions may be guaranteed to run to complete or to cause a resource fault. The number of instructions before a fault may be tightly bounded using polynomial resources to compute the bound.

Outline of Proof: (Existence of Globally Bounded Recursion) Nielson and Nielson [54] present a process algebra for a polymorphic subset of Concurrent ML, a parallel multiprocessing language [55, 56]. With this, they show that the number of processes and interprocessor communications channels associated with a subset of Concurrent ML is finite. They point out that

their analysis is not complete: some programs with finite resource use will be rejected as potentially infinite due to an inability to infer the finiteness of sequences of (unbounded) recursions which terminate. In addition, their semantics admit sequences in which an infinite number of tasks terminate before a finite number begin, again resulting in an erroneous rejection of a finite resource use-case. Substitute bounded recursion into their schema to preclude partial recursive constructs, rendering their analysis complete. The formal language (the subset of Concurrent ML) and semantics constrain all processors and the Concurrent ML to bounded recursion. In addition, the above guarantees that a step-counting function exists, but not that it will be effective in halting a process that is consuming inordinate resources. Therefore, global bounded recursion requires a two-level priority system in which the step-counting function can interrupt the function being monitored. An infinite regression can occur when step-counting functions themselves could consume excessive resources. This may be precluded in building software radios by constraining the step-counting functions to employ a real-time clock. The existence proof is therefore a first-order result (it assumes two levels of priority and only isolated clock faults), not a completely general result. But, it isolates the risk of exceeding resources to the reliability of the real-time clock (versus the millions of lines of code of the radio applications).

Generalizations: The degree to which Theorem 5 may be generalized to asynchronous multithreaded heterogeneous multiprocessors of software radios touches on much current research in computer science. The degree of further generalization is no doubt limited. Communications paths, data sharing mechanisms and process control protocols (e.g. Compare and Swap; Load Linked/ Validate/ Store Conditional [57]) would have to be bounded in a more general setting than [54] for a fully general proof. Topological space models of computation have produced relevant results including necessary and sufficient conditions for wait-free protocols [58], the asynchronous complexity theorem [59], Non-uniform and uniform Iterated Intermediate Snapshot models of wait-free computation [60] and related results. In addition, practical non-blocking primitives can be implemented on contemporary machines like the DEC Alpha, MIPS R4000 and PowerPC [57]. The question of task termination has been proven to be undecideable if even one process can be faulty (e.g. fail-stop [61]). Theorem 5 asserts that a mechanism for ensuring bounded recursive interprocessor communications together with locally bounded recursive software modules guarantees globally bounded resource. This analysis yields an architecture principle for the software radio.

Architecture Principle #1, Bounded Modules: Software radio architectures which limit (software and hardware) control structures to those that constrain partial recursion to the {ISA}-equivalent of bounded recursion will consume bounded resources, possibly entering a known fault condition. Components that conform to this criterion may be called total bounded modules.

***Implications*:** There are many implications to such an architecture principle. Setting resource bounds so tight that the statistical structure of the demand causes them to be exceeded frequently can drive resource use to zero due to resource-use fault conditions. Loss of service from misallocating resource bounds is no less painful on the service provider than other types of deadlock, but it is easy to avoid. In addition, the theorems do not preclude combinatorial explosion. That is, using these constrained hardware and software structures, one may create data sets and instruction sequences with concrete complexity of $O(2^N)$. One is guaranteed, however, that N is

small enough for the exponential resources to be within those allocated in advance. One *must* know the bounds on N in advance in order to establish the time-out criteria for each loop in the process. Such controlled combinatorial explosion will balance the needs of an algorithm to search a large space (e.g. in an equalizer), against the finite resources of the system. It will also guarantee that such algorithms will not consume more than the allocated resources. In addition, although the type of constructs that yield Ackerman's function cannot in general be ruled out, bounded recursion assures that such a construct would yield a resource fault after a specified amount of resource use. Such limits may be proscribed in the type certification process. Thus, any modules that meet the constraints when tested in isolation will not exceed them when inserted into the plug-and-play environment. In addition, the system will not become unstable because of undefined states since all functions are total. Such guarantees help assure reliable, predictable, plug-and-play software radios.

This section has presented an initial examination of the computational properties of software radio functions. The heterogeneous mix of ASICs, FPGAs, DSPs and general purpose computers can be constructed so as to consume predictable, tightly bounded resources. One must constrain the logic to the bounded recursive constructs with prioritized fault handling for exceeding the constraints. The theorems provide an initial outline of a theory of service integrity needed for plug-and-play modules. The constructions from the proofs suggest engineering techniques for applying the theory to development projects. This stability theory must then be combined with queuing theory [16, 26] and other resource management techniques that address the statistical nature of demand in order to provide a robust management of processing resources.

5. Interface Topologies Among Plug-And-Play Modules

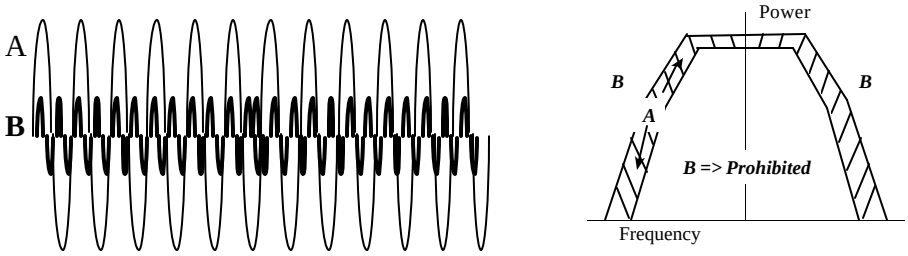
Constraints on the computational structure of modules can ensure bounds on resource use for plug-and-play. Are there constraints on the interface structure among these modules that further facilitate plug-and-play? The answer requires additional aspects of the topological model.

5.1 Topological Spaces

Definition (Topological Space): A topological space, denoted (X, O_x) is a set, X, and a family of subsets O_x, the "open sets", which include X and the empty set, ϕ, and which are closed under union and finite intersection [38]. The topology is the family of subsets, O_x, which has the geometric and algebraic structure.

Topological spaces are very general spaces in which one can represent the geometric properties of interfaces among software radio modules. An interface to an analog source, for example, may be modeled as an infinite dimensional waveform that obeys certain constraints. These include bandwidth, adjacent channel interference, minimum and maximum transmitted power. An uncountable number of such waveforms are possible in an analog interface, but regulatory bodies and the hardware limit the waveform to a structured subset of the possible waveforms. Since the space of randomly perturbed signals is also a metric space [62], one can precisely define distance among waveforms and thus the topological structure of analog interfaces. Figure 14, for example, shows two analog waveforms in the time domain (i); and in the frequency domain (ii). A

range of waveforms is allowed (A) and prohibited (B). Time domain waveform A is within the allowed geometry, while waveform B is not.



(i) Time Domain (ii) Frequency Domain

Figure 14 Interface Waveform Topology [(A) Permitted and (B) Prohibited]

Such widely used interface constraints are ***set-theoretic*** in that they limit the elements in the interfaces to a subset with specified properties. These constraints are also ***geometric,*** defining an interior, A, the conforming region; and an exterior, B, the non-conforming regions. The regions (i) and (ii) are homeomorphic because there exists a topology-preserving map among the two, the Fourier transform.

5.2 Finite Interface Topologies

If a set X has a finite number of elements, |X|, all subsets are open sets (and are also closed sets). If |X| = M, then the number of topologies that induce a topological space on X is $2^{(2\wedge M-2)}$, a double exponential. Not all of the candidate topologies satisfy closure under union and finite intersection as required for a topological space [63]. The huge number of possible topologies compels one to define finite interface topologies more compactly.

Definition (Basis): A set $\mathbf{B} \subset X$ is a basis for O_x if the members of O_x are the union of members of **B**. A basis is a smaller set than O_x from which O_x may be induced by taking unions.

From a hardware perspective, a set of pins in a connector is an interface point, $x_i, \varepsilon X$. The set $\{X = \bigcup_{N} x_I , \phi\}$ is a basis for O_x; but $Y= \{\bigcup_{N\ 1} x_i , \{x_1\}, \phi\}$ which contains three subsets, $\{x_1\}$, ϕ, and the union of all the other interface pins works just as well. The basis with the most subsets is $\{\phi, \{x_i\}\}$, the N+1 sets that include the empty set and each element of X taken as a singleton set. If all the pins are needed for a viable connection, then {X} is the only subset of X that consummates a connection and therefore is the only element in the topology. In this case, $O_x = \{\{X\}\}$; the topology is just the fixed set of required inputs X, which may be called the rigid topology; this is not a topological space because it lacks the empty set. The empty set is not a valid member of the interface set if the interface will not "work" if no pins are present. If the system will "work" with the connector unplugged (e.g. resort to a default or fail-soft mode), then the empty set *is* a member of the interface topology.

From a software perspective, an API may specify a call to the Synthesize() function, for example, with arguments RF, frequency, and W, bandwidth. If both are required, then Y={RF,W} and $O_Y = \{\{Y\}\}$. This is just as inflexible as the equivalent hardware interface. On the other

hand, a tagged API with the expressions RF=859 or W=30 or both or neither would be defined over a space containing {RF}, {W}, X = {RF,W}, and ϕ:

$$O_x = \{\{RF, W\}, \{W\}, \{RF\}, \phi\}$$

The empty set is included in the topology because the interface "works" even if no arguments are provided. Thus, $O_X = \{X, \{RF\}, \{W\}, \phi\}$, the set of all subsets of X or power set, which is a topological space. This power-set topology is the *discrete* topology; it may also be called the *flexible API* topology. This flexible interface geometry is implemented, for example, if Synthesize() uses default values for the missing arguments. Since the power set is the largest set of subsets, the discrete topology maximizes the number of combinations of API parameters over which the function call is valid. The basis for the power set consists of the singleton arguments plus the empty set: $\{\{W\}, \{RF\}, \phi\}$

5.3 Function-call Parameter Topologies

The geometric structure of parameter spaces may be better understood using additional notions.

Simplex: A simplex is an ordered set of points in a topological space that are adjacent in some sense, such as sharing a relation R [7]. Higher dimensionality simplexes induce lower dimensionality simplexes. Simplexes may be embedded in Euclidean space, but need not be [64].

Complex: A simplicial complex is a union of simplexes that includes the union of all the lower dimensionality simplexes of a given simplex.

Q-Connected: Simplicial complexes that share a q+1 face are q-connected [65].

The three vertices of a plane triangle, [A, B, C] in Figure 15, for example, comprise a two dimensional simplex, adjacent in the sense that they are connected by the points in the plane. Each line segment joining these vertices comprises a one dimensional simplex (e.g. [A, C]), the pairs of vertices. A second triangle that shares one line segment with the first is also a simplex. Each triangle together with its edges and vertices comprise a simplicial complex. The two triangles also comprise a simplicial complex in which the simplexes are 1-connected by the line [A, B].

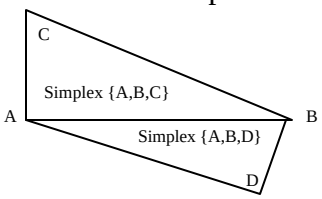


Figure 15 Simplicial Complex Consists of Two Simplexes

[7] Strictly, an n-simplex is the convex hull of n+1 independent points in m-dimensional Euclidean space, which is the intersection of all convex sets containing those points subject to completeness and closure axioms [Kahn, 95]. This level of geometric precision is not needed for this software radio application.

Interface parameter sets may be modeled as simplexes with a large number of dimensions. In the Synthesize() function above, suppose RF is discrete and has as its explicit basis the range [400.001, 2400.000] MHz and resolution 1 kHz. The RFs that are valid parameters of Synthesize() are adjacent in sharing the relation R = "valid for Synthesize()". They therefore constitute a simplex consisting of the two million distinct RFs in this range. Let BW be defined for {25, 30, 200} kHz. The parameter space {RF, BW}, contains 6 million points, a simplicial complex consisting of the union of three simplexes of 2 million RF points each, indexed by BW. Then the set of RFs reachable from Synthesize(925.000, 30) is exactly one, {[925, 30]}. Synthesize(BW=30), on the other hand, reaches the two million point RF simplex, e.g. through a global default. Synthesize() reaches all six million points through defaults, connecting them in a single simplex.

For the example function Synthesize() to have an explicit basis, its definition must include basis tuples expressed as Parameter(Type, Range). Parameter indicates the parameter being defined. Type chooses among predefined subspace types (e.g. Discrete, Continuous). Range sets the valid values. In the example above, an explicit basis could be BW(Discrete, {25, 30, 200}) and RF(Discrete, [400.001, 400.002; 2400.000]) using Mathcad® range notation. This explicit basis can be enforced by the environment to yield an error result "Out of Range" for arguments that violate the basis description. Such semantics define Synthesize() as an effectively computable function over all inputs. A product topology on these bases yields the six million points of (BW x RF) as the input space to this function. However, it is often necessary to limit the parameters to subsets of this space. One might limit the 200 kHz bandwidth to the RFs allocated to GSM, say, 925 to 960 MHz. The simplex

NOT (200, {[925.000, 925.200, 926.000]}) -> 'Bandwidth-out-of-range'

defines the set-complement subspace for which the 200 kHz bandwidth choice is mapped to a fault condition. To the degree that such subspaces can be changed while the radio node is in the field, the service provider has a "future proof" interface. In order to be changed in the field, the "capability" implicit in the interface constraints must be defined in such a way that the control algorithms can test and manipulate the constraints, e.g. to decide whether to accept a download. The geometric structure of the interfaces naturally defines plug-and-play constraints.

5.4 Plug-and-Play Interface Geometry

An implementation conforms to its own as-built interface (whether documented or not). Historically software radios use a mix of custom internal interfaces not designed for plug-and-play. The implementation decisions made during design, development, integration and test constrain the interface to some specified point within the space of interface topologies as illustrated in Figure 16. An arbitrary interface simplex consists of all the parameters, signals, etc. related to an interface, ranging over the (possibly infinite) set of possible values of each parameter. The arbitrary interface in the figure is the simplicial complex consisting of all subsets of the arbitrary simplex. Each domain and range of a plug-and-play interface, however, consists of a designated subset of this simplicial complex. A plug-and-play interface defines an interoperable subset (not just a single point) of the interface space. These subsets include physical and logical interfaces (e.g. defined in the APIs and interface control documents). The physical interface subspaces must change as a function of the hardware in which the service is delivered. The logical interface sub-

spaces also may have to change as a function of the software modules configured to deliver the services. Independent evolution of software and hardware requires interface geometry that is independent of whether the interface is purely hardware, purely software or some mix of the two evolving over time. Extensible plug-and-play constructs a capability dynamically. To do this, the control system must have a way of comparing the range of one function to the domain of the next to determine whether the functions are compatible.

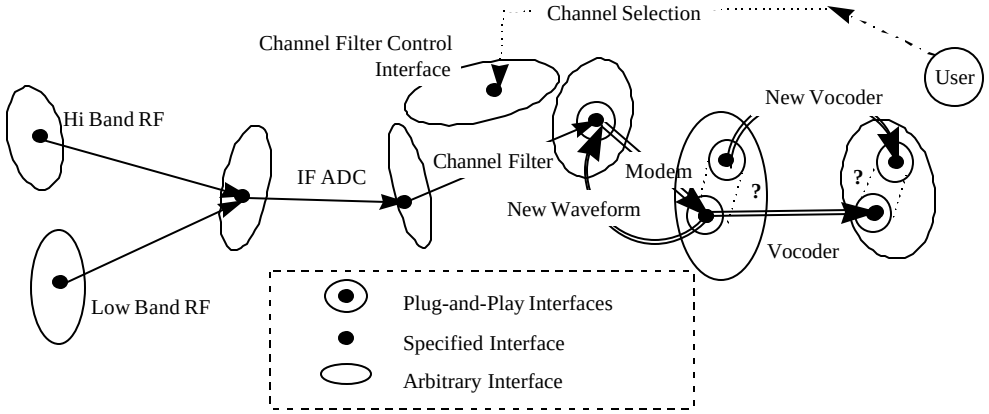


Figure 16. Interface Geometry Reflects Design Decisions

5.5 Extensible Capabilities

Plug-and-play modules -- hardware or software – should be accepted into a system only if it has the capability to support them. Capabilities may be represented as levels (or "tags") such as the type of video teleconferencing interface in the ITU H.320 Recommendation [66] or the type of call control support from Microsoft's Telephony Applications Programmer Interface (TAPI) [67]. The advantage of the tag-based approach is its simplicity. Capabilities are defined during the standards-setting process and documented in the text of the standards. One disadvantage of this approach is the possible misinterpretation of the text of the standard. To counter this chronic problem in telecommunications standardization, ETSI has promulgated language that would make the formal specification of the ITU Z.100 Specification and Description Language (SDL) [68] the *normative* expression of the standard [69], with text providing amplification and explanation. Historically, the text has been normative while the computer-processable representations have been auxiliary. Another disadvantage is that levels of capability cannot be defined dynamically by the radio system. Nor can they be manipulated computationally by the joint control system of the radio infrastructure. For control of dynamically defined software radio-based services, the constraints must eventually become dynamically computer-processable. That processing must support automated reasoning over the constraint spaces and capabilities of the host system so that the radio itself can construct the level of capability necessary for the specific plug-and-play services. SDL falls somewhat short of this vision since semantics are left to the designer. Dynamic capability definition requires designer-independent semantics.

The input and output spaces of an ADC, for example, might be as illustrated in Figure 17. The list of features defines the dimensions of the interface space. Range expressions define subspaces. Discrete values constrain a subspace to a single point. These spaces often need to be

extended. Parts that are defined a-priori may be extended dynamically. For example, to add a control that selects from either the A or B buffer in a double buffered interface of this notional ADC requires extension of the interface topology, adding sets and subsets to (X, O_X).

Resource:
- **Object Type:** ADC
- **Domain (Input Space):**
  - **Class:** Analog-Stream
  - **Family:** Coax-DC-Coupled
    - **Impedance:** 50 Ohm
    - **Connector:** BNC
  - **Carrier Frequency:** Baseband
  - **3dB Bandwidth:** 350 kHz
  - **Signal:** {Analog, Predetected-Signal, Band-Limited-Signal}
  - **Control Subspace:**
    - Gain: [0dB to 20 dB, 1 dB steps]
    - Gain Control: AGC
  - **Dynamic Range:** 70 dB
- **Range (Output Space):**
  - **Class:** Digital Stream
  - **Hardware Family:** Serial-ECL
    - **Clock:** 15.44 MHz
    - **Framing:** 544 kbps
  - **Sampling Frequency:** 1 MHz
  - **Nyquist Bandwidth:** 500 kHz
  - **Signal:** {Digital Signal, Predetected Signal, Band-Limited Signal}
  - **Resolution:** 16 bits
  - **Dynamic Range:** 65 dB

Figure 17 Topological Space Representation of an ADC

It is easy to do this in a way that is readable by people. To do this in a way that is processable by a control algorithm requires a Radio Knowledge-Representation Language (RKRL) as suggested in the example of Figure 18. The meta-level expressions <Buffering>, <Buffer-Size>, <Buffer>, and <Buffer-flag> introduce the primitives **Buffering, Buffer-Size, Buffer** and **Buffer-flag**. The RKRL is assumed to include a-priori semantics for Resource, Type, Set-Extension, Output-Space, Range, None, Double, and Singleton topologies. The set extension augments the interface topology (X O_X) with new dimensions. The syntax of the extension indicates that if **Buffering** = None, then the effect is null. This effectively glues this new subspace to the existing subspaces in which there was no buffering. In addition, the new subspace **Buffering(Double)** has the rigid interface topology (e.g. this device has a double buffers A and B of 128 Bytes each). The **Buffer-Flag** has a singleton topology over interface port '2A1' that would be set to the value zero or one. Topological spaces naturally express constraints geometrically. Such an approach represents radio capability in a way that can be processed by control algorithms.

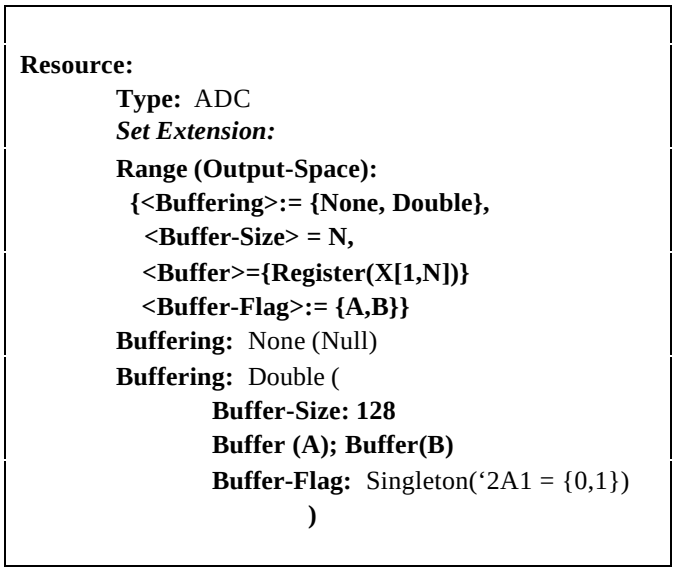


Figure 18 Resource Topology Extension: ADC Buffering

The considerations involving interface topologies may be expressed in the following architecture principle:

Architecture Principle #2: Explicit Extensible Interface Topology: API's and hardware interfaces which exhibit an explicit basis for each interface parameter space and which are extensible in the field exhibit a level of flexibility and adaptability suitable for extensible plug-and-play applications.

Implications: The requirement for an explicit basis assures that the interface topology is defined completely, including error states. Each of the key software radio interfaces defined by the top level functional component maps has a corresponding set, X, and family of subsets, $O_{x,}$ which characterize the interface. The full extensibility of the topological bases in the field requires some kind of RKRL-based representation of radio resources. This is an area of current research in wireless computer-communications systems [70].

6. Architecture Partitions

One goal of a plug-and-play software radio architecture is to provide computing resources from as yet undefined hardware modules to support as yet undefined software modules for as yet undefined services. The following architecture analysis examines the functions, components and design rules of software radios towards this goal. This analysis applies the computational and topological properties introduced above to begin to identify the mathematically based partitions of software radio components. The approach initially examines the source code of a widely published military software radio, SPEAKeasy I [71]. A sequence of virtual machines is then derived that both expand the {RAM} equivalent instruction set and constrain the computational geometry, yielding a layered virtual machine reference model for the software radio.

6.1 SPEAKeasy I

SPEAKeasy I resulted in the software radio source code structure summarized in Table 3.

Table 3 SPEAKeasy I Software Modules

Module	**Source**	***Ada Module Descriptions/* Functions**
At	(127 kB)	C040 interprocessor communications
BIT	(318 kB)	Built-In-Test packages, including CRC, EEPROM, PID, I/O registers, interrupts & DMA
Cm	(1.29 MB)	Configuration Management
ALE	(125 kB)	ALE Receive (Rx) and Transmit (Tx) Functions
ALE_Rx1	(378 kB)	Automatic Link Establishment (ALE) Receive Modules
Hvq	(645 kB)	Have Quick Communications Ensemble
Hvq_Ct	(109 kB)	Control Modules (Initialization, Mode Control, Errors)
Hvq_Glob	(25 kB)	Globals
Hvq_ Rx	(379 kB)	Receive Mode (Synchronize, TOD, Rx, Active...)
Hvq_ Tx	(131 kB)	Transmit Mode
Work	(299 kB)	ALE packages & specs
Hfm	(518 kB)	HF Modem Communications Ensemble
Hfm_ctrl	(58 kB)	Controls Waveform start/stop messages; RTS Events; PM Query; TX/RX Done (local);
Hfm_dc	(22 kB)	Data Control Packages, source messages error checking
Hfm_ rx	(289 kB)	Receiver Bit & message operations, text I/O, Rx utilities, data correlation tables, filters, queues
Hfm_ tx	(149 kB)	Squelch, TX/RX mode, TX templates, RF Control, Timing
Nbg	(334 kB)	Narrowband Frequency Hopping
Nbg_ct	(49 kB)	State Machine, Sync Loss, TX/ RX, Waveform, PTT State...
Nbg_glob	(105 kB)	Global parameters for NBG package
Nbg_hp	(57 kB)	Hop Packages – timing, data request/processing, PTT acknowledge, cryptographic processing...
Nbg_rx	(73 kB)	Receiver Packages MFSK, Preamble, Galois (FEC), Dead Bits, Flags, Bitsync, RX flush, Detect/Track
Nbg_t	(49 kB)	TX: Amplitude, Preamble fill, IQ Samples, AM on Voice, Filter, Inter-Process Communications (IPC) Messages, SSB, DSB, QAM, OQPSK, Event & Constraint Checking

The as-built code has some strong features - such as real-time performance and accurate handling of timing differences between radio networks. Since an Ada implementation was mandated, the real-time executive is the Ada run-time kernel. The Ada modules can be viewed as software radio objects. They include databases, channels, and agents. Databases store personalities; filter parameters; lengthy chunks of compiled code; and data sets to be loaded into a personality at run-time. Channels are abstractions around which modes (e.g. HAVE QUICK) are organized. A channel is supported by several Ada packages that perform the systems level functions of RF control, modem processing, INFOSEC, and related internetworking. A channel:

(1) gets the system resources (paths or threads through the system);
(2) installs its personality on these resources to implement a mode; and then
(3) keeps track of the overall state of the processing thread that delivers the associated services.

In SPEAK*easy* I, lower level modules implement the personalities of the channels. They also serve as hosts for busses, manage IO processes, access timing and positioning data, and control the radio. Timing packages manipulate the system clock, Time of Day (ToD), the day/time format, and the timing resolution. The RF control packages determine RF direction (i.e. transmit or receive); the RF mode (e.g. linear or nonlinear amplification); pre-emphasis for pre-distortion; and frequency of transmission. The modem packages include modulation (AM, FM, QAM, USB, MSK ...), demodulation, automatic gain control (AGC), loop bandwidth control, and data packing and unpacking of protected bits. Channels also keep track of the status of the mode, number of resources employed, volume, data rate, throughput, network parameters such as network number, and assigned time slot(s). The system may also be asked to perform a loop-back function for network testing or local diagnosis. The back-end functions include message processing, internetworking, managing protocol stacks, and the user interface. Radio control handles system boot-up, initial ToD, current hop, calibration, status requests and security level. Comparing the Nbg, Hvq and Hfm mode software shows a common pattern of control module(s), global parameters, transmit (modem), and receive (modem) modules along with specialized modules such as hop generation for frequency hopping modes.

The as-built code includes four distinct types of software: services such as voice, data, and channel bridging; software that structures the radio applications such as state machines; infrastructure that moves data; and hardware-specific support software such as the operating system. About 30 to 40 % of this code is infrastructure, 30 % structures radio applications and the remaining 30 to 40% implements services. These different functions are almost inextricably intertwined. Much of the code is hardware-dependent. The partitioning analysis that follows is oriented towards reuse and therefore begins at the most primitive level of existing "code", the hardware.

6.2 Hardware-Specific Partitions

The hardware interfaces define the lowest level physical partitioning of a radio system. From a topological perspective, a digital processor *connects* the states of data in its registers. The set of data states reachable in one clock cycle from any initial state constitutes the processor's characteristic simplex. Let [RAM] indicate the characteristic simplex of a processor with instruction set architecture {RAM}. A processor with twenty 32-bit address registers, cache pointers, general purpose registers, DMA registers and other registers can in principle assume any of $2^{(32*20)}$ states. The number of states equals $10^{192.6}$ or 0.358846241 M 2 (see the Appendix for magnitude notation). This is the size of the processor's instantaneous characteristic simplex. Fewer registers yield a smaller simplex. A more complex instruction set architecture yields a larger simplex. An FPGA's simplex is determined by its use of state memory, which will generally be much larger than a DSP chip of the same area and device feature size. The size of the simplex indicates how many different things the processor can accomplish in a single clock cycle.

6.2.1 Paths in Simplexes Induce Partitions

Sequential simplexes are connected to each other by the clock sqeuence. Over time, a series of RAM instructions traces a path through a sequence of simplexes, the union of which is a simplicial complex (SC). In one second, a processor with a 100 MHz instruction clock may assume any of $(10^{192})\wedge(10^{8})$ or 0.87822M5 states. In addition, a software radio with distributed multiprocessor hardware and interconnect devices has additional connectedness among processor simplexes as illustrated in Figure 19.

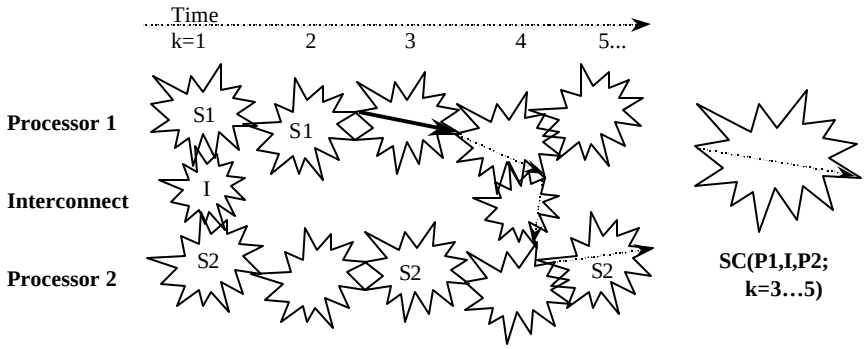


Figure 19 Processor Simplexes (S1, S2), Interconnect Simplexes (I) and an Equivalent Simplicial Complex, SC(P1,I,P2; k=3 ... 5)

At each clock cycle, an edge in the simplex is traversed (e.g. the solid arrow in Processor 1's simplex at time k=3 in the figure). The path traversed through a sequence of simplexes may be subsumed into an SC, e.g. SC (P1, I, P2; k-3 … 5) in the figure. The dotted arrow in SC indicates the path traversed by data sent from P1 to P2 during clock sequence [3, 5]. The way in which software constrains such paths implicitly defines architecture partitions. A larger number of possible paths indicates greater built-in flexibility while topological loops in such paths define the natural partitions of the software.

6.2.2 Interrupt Service Routine (ISR) Topological Loops

Consider hardware-dependent software such as operating system services, which consumes processing resources in a way that is tightly coupled to the hardware-related processor states. A typical ISR, for example, might turn off hardware interrupts, push the processor state onto a stack, test the interrupt condition, set a dispatch pointer, restore the registers and exit. In a real-time system, this process could take from 10 to 100 instructions. Windows NT requirements for template data might expand this to 1000 instructions or more. During this sequence, the machine's simplicial complex, normally available for arbitrary computing, is constrained to a path that involves states of a particular control register (the interrupt register). In addition, the instruction sequence that is so constrained is defined temporally by the start of the interrupt and the termination of the ISR. This produces a path in the simplicial complex that begins with a hardware interrupt "set" and a particular configuration of registers and ends with the interrupt "clear" and the identical configuration of registers. Define a convenient distance measure on data states such as Hamming distance, the number of bits that are not identical. A topological loop over

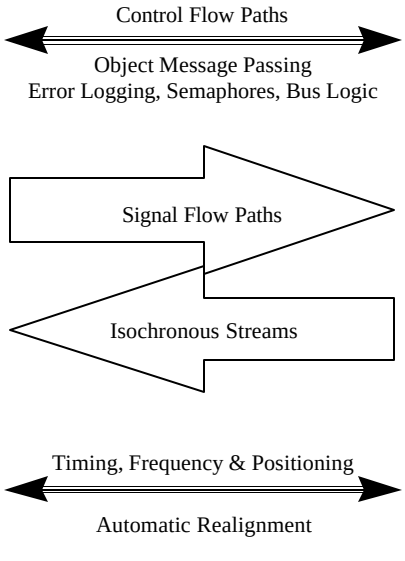
([RAM], d) at x, a point in [RAM], is the shortest path in SC[RAM] that returns to within d of x if one exists. An ISR is a topological loop in [RAM] for d=1 in the above example. Input / Output drivers, task schedulers, disk access control, real-time clock support software and the like are other examples of such hardware-specific software included in operating system kernels, which have short characteristic topological loops.

6.2.3 The Topology of the Kernel Substrate

The set, {kernel}, comprises the lowest level software substrate. Kernel sequences can be modeled as extensions to the processor's native ISA. The hardware interrupt ISR, for example, could be modeled as the function Service() defined over the set {Interrupt-Registers}. Service() is an instruction in {kernel}. From a topological perspective, Service() defines a sub-graph within the processor's SC. A processor with a library of such hardware-dependent modules can be viewed as a machine with an extended instruction set, ISA^{+} = {RAM} union {kernel}. The set {kernel} may have a distinguished sequence of {RAM} instructions for the Idle state. Idle ⇔ {RAM} in that during the Idle state, an arbitrary RAM sequence is possible. Let {kernel'} represent the kernel less Idle. Minimizing the number of {RAM} sequences needed for the {kernel} maximizes the {RAM} capacity left for higher-level radio services. The simplex of {RAM} union {kernel'} is no larger than [RAM], but it contains a distinguished subspace corresponding to the paths reachable through {kernel}.

6.3 Infrastructure Software Topology

Assume that each processor in a software radio has an associated set of kernel software. The next step in a bottom-up process of defining virtual machines identifies those functions that allocate, set up and control the physical resources to create logical resources. Figure 20 lists the function calls required for distributed processing. These functions include the structures recommended in ITU's X.900 Open Distributed Processing reference model [72].



Initialize, Check-In, Shutdown,
Create Port, Find_Port, Delete_Port
Get_Message, Send_Msg, Send_RPC(timeout)
Multicast_Register/Send/Release/Echo
Error(string), Extract_UDP_Socket/_IP/_Port

Load_Path, Load_Command_List
Path_Get_Status/_Type/_Size/_Cmd
Path_Echo,
Path_Open, Path_Close,
Path_TxInit/_TxReset/_TxSetID/_TxLoad/_GetPTR
Path_RxInit/_RxReset/_RxSetID/_RxSetMask/
/_RxContinuous/_RxSnapshot/_GetPTR/_Status
Path_Catch, Path_Throw (UNIX/VxWorks)
Device Specific (PCI, VME, DT, Transputer...)

Get_Time, Get_Drift (Inernal vs. GPS)
Clock_Status, Clock_Terminate (Port)
Clock_Realign(Parameters)
Send_Sync_Manager
Init, Reset(error), Status, Trigger(parameters),
TimeUp, TimeDn, TimeEqual, TimeLess, TimeGT
Shutdown, GetRate, FreeList(FIFO), Load, Queue
Clear, Test, TestStatus

Figure 20 Infrastructure Software Function Calls

They also include structures unique to radio applications such as frequency distribution. This infrastructure code manages control flow paths; signal flow paths; and timing, frequency and positioning information. It also includes features needed for isochronous delivery of voice, video and other real-time streams. The control flow paths mediate message passing among most objects in the system. Error logging; semaphores that manage shared resources; and bus access protocols are examples of control message flows. The infrastructure functions initialize the system, create and manipulate logical and physical ports, move messages, and perform remote procedure calls (RPC). They also provide multicast services, handle error messages and support standard protocol interactions such as Internet Protocol (IP). Multicast simplifies programming of multichannel operations such as initializing 100 subscriber channels distributed among 25 DSP chips. The control-flow methods listed in the figure constitute a minimum set necessary for software radio infrastructure.

The signal-flow methods listed in the figure, similarly, set up and manage signal flow paths among processes on the same or on different processors. These isochronous streams must meet specified timing constraints. Due to the overhead associated with path set up and tear down, these paths must be opened and closed multiple times without being set up and torn down again. Timing, frequency and positioning is a very involved process, a complete discussion of which is beyond the scope of this treatment. Time references obtained during network synchronization must be maintained on a per-network basis. Since the software radio generally participates in multiple networks simultaneously, it must maintain absolute time per network. This is accomplished not by changing the software radio's clock, but rather by defining time offsets for each network. Additional ancillary functions related to queuing data messages (e.g. load, clear, test status, and reset) are also part of infrastructure.

Applying the virtual machine paradigm to this set of software yields a new virtual instruction set {Infrastructure} which consists of {Message-Passing}, {Isochronous Paths}, {Timing}, {Frequency} and {Positioning} instruction subsets. These extensions build on the facilities of {kernel}. The {RAM} simplex contains {Message-Passing}, etc. Each such subspace depends on {kernel} services, the lowest layer of the emerging virtual machine hierarchy.

6.4 Radio State Machines

State machines control access to many software radio resources. State machines typically control transmit and receive channels and fault recovery actions. A resource-allocation state machine is illustrated in Figure 21. Three states are shown in boxes: waiting for instantiation, fetching a waveform and waiting for a response. The arcs are labeled with conditions that cause one to transition from one state to the next and with actions performed upon such a transition. This control structure may be viewed as a software object with a set of control slots (the states) and attached methods. Methods test for state transitions and perform required actions. Recommendations for defining and simulating such state machines are provided in the SDL Recommendation Z.100.

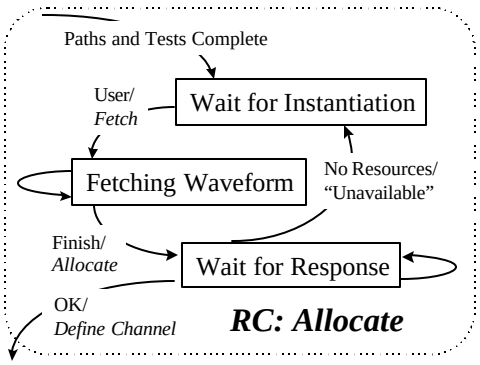


Figure 21 Infrastructure State Machine

Such state machines may operate on several levels. At the top level, each channel (e.g. HAVE QUICK) has a top-level channel-setup state machine that keeps track of resources. The next-level state machine manages the mode, as illustrated in Figure 22. The channel is initially idle. When it has been set up, it enters the Ready state. If the user (or network) initiates a "talk" sequence, the transmit (Tx) state is entered, upon which the radio will alternate between Tx and Rx states until conditions are met which either suspend or deactivate the channel.

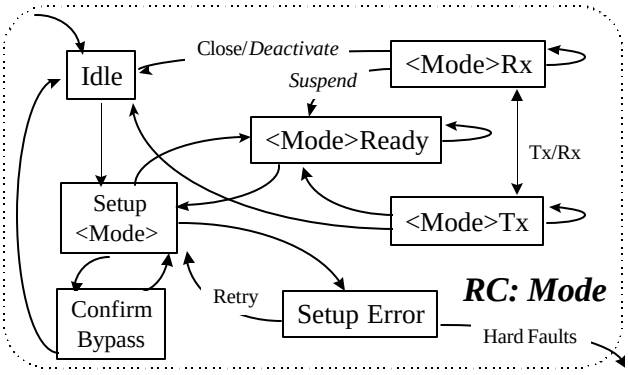


Figure 22 Channel Control State Machine

Lower level modem state-machine tracks states for active plain text receive and transmit; crypto-sync; receiving analog and digital; and bridging fades. The SPEAKeasy state machines also have built-in timing and error-recovery procedures. Automatic Gain Control (AGC) and Squelch (Sq) are adjusted on all transitions in the lower-level state machines. Thus, the state machines schedule both routine processes such as squelch and AGC and processes driven by channel conditions such as bridging across fades. States also reflect failure modes. These include sync failure, loss of carrier and loss of system resources. Such state machines are a central mechanism for controlling system and radio resources in the software radio.

From a geometric perspective, the topological loops of these state machines begin and end on conditions of identical state values. That is, the states "set-up", "transmit", "receive", etc. are the constants or fixed points in the evolution of the [RAM] simplicial complex. Periodically, the processor's register set will return to topologically close states for some suitably defined distance

d. Since the functions called from these state transitions (e.g. “transmit next block”) employ the facilities from {Message-passing}, {Isochronous stream}, etc., these state machines define a new level of virtual machine built on top of the Infrastructure machine. The boundaries of this layer of virtual machines, then, may be based on the function calls attached to the state machine objects. This new layer of virtual machine is the {State machine} instruction set, consisting of the {RAM} sequences that set up, execute and control the state machine objects. The state-machine objects themselves comprise a virtual processor architecture

6.5 Channel Agents

The characteristics of SPEAKeasy I, SPEAKeasy II and other software radios have been abstracted to create the software object diagram of Figure 23. These high-level objects may be called channel agents. Each software object has been allocated functions from the radio functional block diagram of Figure 2.

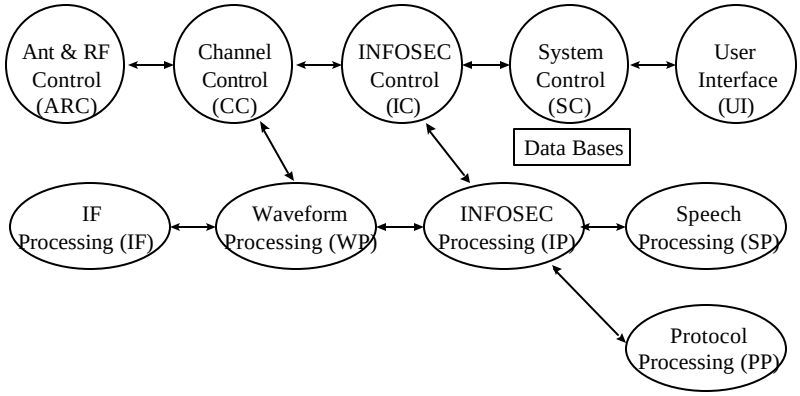


Figure 23. Channel Agent Software Objects

Those agents in the top row of the figure control the system while those in the lower rows implement the traffic channels and related services. Each of these high-level software objects has subordinate objects, to the level of primitive single-function radio objects such as filters, modulators, interleaving, clock recovery, and bit-decision objects. Some of these may be implemented in FPGA personalities, but most would be DSP code. The protocol and speech processing “back end” of the software radio employs a protocol stack such as ATM, TCP/IP, Mobile IP, etc. Consequently, internetworking to the wireline infrastructure consists of a few monolithic predefined software objects. The objects of Figure 23 implement the functions summarized in Table 4.

Table 4 Functions of Radio Software Agents

Radio Agent	**Object Methods and Slots**
Antenna & RF Control (ARC)	TX/RX, Power, Polarization
Channel Control (CC)	Allocate Resources, Configure, State Machines
IF Processor (IF)	Diversity combining, Beamforming, Pre-emphasis
Waveform Processor (WP)	Generation, Timing, Fault Detection, Modulator and Demodulator (Modem), link-related data processing
INFOSEC Control (IC)	Key, Control Bridge to Black Side, Authenticate
INFOSEC Processing (IP)	Encrypt, Decrypt, TRANSEC

System Control (SC)	Initialize/ Shutdown, Test, System Status
User Interface (UI)	Commands & Displays
Speech Processing (SP)	Codecs; Echo Cancellation, Voice Channel Modems
Protocol Processing (PP)	Packetization, Routing

These radio functions are relatively generic. Thus, each waveform or mode of operation may be expected to have either its own set of agents or a set of parameters and a script that tell the agent how to treat that specific mode. Radio services, like bridging across waveforms, would be constructed as scripts or Java-like Applets that interconnect Channel Agents and other high level functions. The semantics of such top-level modules are readily represented in the Unified Modeling Language (UML) [73]

The timing and overall behavior of the channel agents are constrained by the lower level state machines that essentially tell the agent what is going on in the channel. Thus, from a geometric perspective, agent subspaces in the [RAM] simplex subsume the {State machine} subspace. The extended instruction set for channel agents {Channel agents} is more of a category than a virtual instruction set. The specific functions {Antenna and RF Control}, {Channel Control}, {IF Processor}, {Waveform Processor}, etc. each constitute a set of virtual instructions. Since these channel agents must be mutually consistent in order to work together, the class {Channel agents} refers to a mutually consistent set of such instruction set extensions.

6.6 Distributed Layered Virtual Machine Reference Model

The preceding sections have described essential software mechanisms employed in the construction of software radios. When viewed in terms of the subspaces defined by the paths traced in the simplex of an equivalent RAM processor, layers of successively less machine-dependent software emerge. The distributed layered virtual machine is a way of representing the resulting hierarchy. Each layer is a virtual machine built on subordinate layers. The virtual machine interfaces may be held constant so that the implementation details and intellectual property present within a component at a given layer are hidden from all other layers. The software radio architecture that results from this process is illustrated in Figure 24.

As the capacity of FPGAs and DSPs continues to grow, the processing to support such virtual machines becomes more affordable. The services offered from the top-layer virtual machines thus become independent of the hardware implementations in the bottom-layer virtual machine. The FPGA and DSP hardware vendors, for example, could offer infrastructure software that supports radio applications. This layer could be implemented in part using an augmented Message Passing Interface (MPI) [74]. The state machines in the infrastructure and radio applications layers would naturally fall into the domain of SDL. Third party waveform vendors could offer middle layers. Interfaces among components may be represented in UML or the Interface Definition Language (IDL) [75], possibly augmented with Abstract Syntax Notation (ASN.1) [76]. Systems integrators and service providers could then define their unique value-added in the top layers, building on the broad base of industry support at the lower layers. In addition, Computer Aided Software/ Systems Engineering vendors could embrace the wide range of reusable components to assist developers to encompass the entire software radio distributed virtual machine hierarchy for reduced time to market.

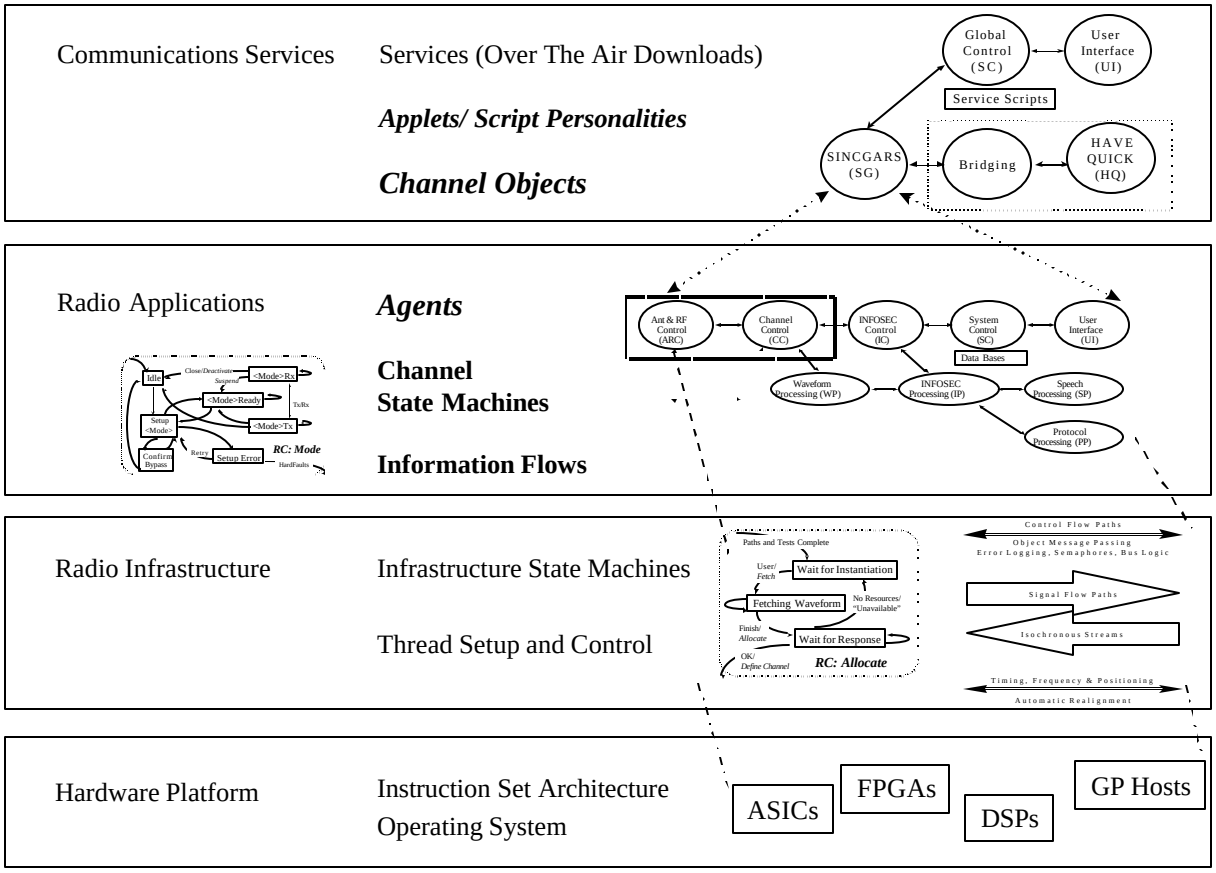


Figure 24 Software Radio Distributed Layered Virtual Machine

These observations may be summarized in a third architecture principle:

Architecture Principle #3: Distributed Layered Virtual Machine Reference Model: Modules implemented according to the distributed virtual machine reference model will insulate lower layer hardware-dependent modules from upper layer service-defining modules. In addition, the topology of the interfaces can guide the further refinement of this initial model.

The economic incentives for widespread adoption of such a model have to do with integrating markets for economy of scale. The global wireless marketplace now consists of dozens of niches defined by unique hardware platforms and waveform-unique infrastructure. Industry is attempting to organize itself to integrate these diverse markets so that the next generation of wireless will offer low-cost plug-and-play services. The process of defining the required industry standards has already begun. The MMITS forum, for example, has defined an architecture framework based on functional threads in which a generic Applications Programmer Interface (APIs) is being defined across the virtual machine. MMITS is integrating the GloMo Radio Device API [77] into its own. The Object Management Group (OMG), on the other hand, has requested technology for CORBA real-time objects [78] including multimedia at video data rates. A real-time CORBA could provide software radio infrastructure. The partitioning strategy and layered

reference model presented here may provide insights helpful in evolving towards broadly accepted standards for future software radios.

7. Conclusion

The computational and geometric properties of the software radio will determine the degree to which plug-and-play is realized in the marketplace. Greater understanding of the mathematical properties of software radios should accelerate progress toward cost-effective plug-and-play services to which developers aspire. In particular, the architecture principles are revealing:

1. **Bounded Recursive Modules**: Construct the software radio system -- host environment and plug-and-play modules – of bounded recursive modules. These modules will be defined for all inputs; and they will consume predictable system resources. Although the software may contain errors, those errors will be much less likely to cause faults in the delivery of audio and multimedia streams; they also will be less likely to cause system crashes than unconstrained modules.
2. **Explicit Extensible Interface Topologies**: Define software radio interfaces using an explicit basis for the underlying topological spaces. Use extensible languages such as UML, SDL, IDL and ASN.1 until a more general language such as a Radio Knowledge Representation Language (RKRL) emerges.
3. **Distributed Layered Virtual Machine**: Position plug-and-play modules in the distributed layered virtual machine hierarchy in a way that maximizes the use of existing (lower and higher level) components.

Appendix
Magnitude Notation

A recursive logarithmic number system makes the large size of simplexes and simplicial complexes easier to understand and manipulate. The notation yMn ("y Magnitude n") indicates raising y to the power of 10 n times where n is an integer and $0 < y < 1$. The magnitude, n, of this measure of concrete complexity of a simplicial complex is a strongly nonlinear measure of complexity. For example, $32 = 10^{1.5051}$ which equals $10^{\wedge}10^{0.1775}$, which can be expressed in magnitude notation as 0.1775M2, a magnitude 2 number. The size of the processor's simplex above, $10^{192.6}$, equals 0.35884M3, a magnitude 3 number. The one-second simplicial complex has size 0.87822M5. This magnitude 5 number is exponentially bigger than a magnitude 3 number by 10^{10}. Put another way, the size of this complex is 10^{10} raised to a power containing 35 million zeroes. For a perspective on the size of this number, the number of neutrino-sized places in the known universe (10^{40} meters on each side) over the last 10 billion years is only 10^{266} or 0.384836241M3. Using the size of the simplex ($10^{192.6}$) as a base and raising it to the power k = 100 million is not, perhaps, a familiar concept, but it establishes an upper bound on the size of the simplicial complex associated with just one second of use of such a processor. It is useful in quantifying the architecture's effectiveness in control of combinatorial explosion.

---

1 McGarth et al, "RFIC Technology for Wireless Consumer Products - Trends in GaAs", *M/A-COM LOUD & Clear*, (Lowell, MA: M/A-COM, Inc) 1995

2 Mouly and Pautet, *The GSM System for Mobile Communications* (Plaiseau, France: Published by the authors, 1992)
3 Upmal and Lackey, "SPEAKeasy, the Military Software Radio" IEEE Communications Magazine (NY: IEEE Press) 1995
[4] Nicholson, D. Spread Spectrum Signal Design LPE and AJ Systems (Rockville, MD USA: Computer Science Press, 1988)
[5] Reference Data for Engineers (Carmel, Indiana: Sams Publishing, 1993)
[6] Stallings, Handbook of Computer-Communications Standards, Volume1, The Open Systems Interconnection (OSI) Model and OSI-Related Standards (NY:Macmillan, 1987).
7 Pickholtz and Hill, Adaptive Beamforming for Interference Reduction, GW University PW3312A, 31 Dec 1990
8 Zoltowski et al, "Blind 2-D Rake Receivers Based On Space-Time Adaptive MVDR Processing for IS-95 CDMA System" MILCOM 96 (IEEE, NY) Oct 96.
9 Belzer et al, "Joint Source Channel Coding of Images with Trellis Coded Quantization and Convolutional Codes" (Los Angeles: UCLA) 98
10 Ferguson and Huston, Quality of Service (USA: Wiley) 98
11 Paradells et al, "DECT Multibearer Channels" 0-7803-1927-3/94 (NY: IEEE Press, 1994)
12 Strom and Shaula, "Optimistic recovery in distributed systems", ACM Transactions on Computer Systems (USA, Association for Computing Machinery) 1985
13 Pesonen, "Object-Based Design of Embedded Software Using Real-Time Operating Systems" IEEE 1068-3070/94 (NY: IEEE Press, 1994)
[14] Tebbs & Garfield, *Real Time Systems* (Berkshire, UK: McGraw-Hill) 1977
15 Ellison, K., Developing Real-Time Embedded Software in a Market-Driven Company (NY: Wiley, 1994)
16 Mitola, J., "The Software Radio Architecture", IEEE Communications Magazine (IEEE, NY), May 95
[17] Mitola, J., *Software Radios: Wireless Architectures for the 21st Century* (Fairfax, VA: Statis*faction*) 97
[18] MMITS Forum Technical Report (www.mmitsforum.com) 1997
[19] Milewski, *The Topology Problem Solver* (Piscataway, NJ: Research and Education Asociation) 94
[20] Herlihy and Shavit, "A Simple Constructive Computability Theorem for Wait-free Computation", STOC 94, (Montreal, Quebec, Canada: ACM) 94
[21] DII Strategic Enterprise Architecture, DISA, 1994 IEEE Communications Magazine, (NY: IEEE Press) Oct. 95
22 Programmable Digital Radio (www.mmitsforum.org: GEC Marconi) 1996
23 The Software Defined Radio Request for Information, (Atlanta, GA: BellSouth) Dec 1995
24 Pengelly, R. "Low Cost GaAs MMIC Chip Set for Dual Mode AMPS/CDMA Phones", WESCON 96, p 96
25 Oh, S. "DSP Technology for a Hands-Free Car Cellular Phone", WESCON 96, p 821
26 Mitola, J., "Software Radios: Technology and Prognosis", Proc. National Telesystems Conference (IEEE, NY), May 92
27 Walden, "ADC Integrated Circuits", Proceedings of the IEEE GaAs IC Symposium (IEEE Press, NY) Dec 95
28 Zangi and Koilpillai, "Software Radio Issues in Cellular Base Stations", JSAC on Software Radios (NY: IEEE Press) 1998.
29 Pickholtz and Hill, *Adaptive Beamforming for Interference Reduction*, GW University PW3312A, 31 Dec 1990
30 Loundu et al, "Estimating the capacity of a frequency-selective fading mobile radio channel with antenna diversity", 0-7803-1927-3 (NY: IEEE Press, 1994)
31 Morgensen and Petersen, "Practical Considerations of Using Antenna Diversity in DECT" 0-7803-1927-3/94 (NY: IEEE Press, 1994)
[32] Tsui, j., Digital Techniques for Wideband Receivers (Boston: Artech House) 1995
[33] Abidi, "Low Power Radio Frequency IC's for Portable Communications," Proceedings of the IEEE (NY:IEEE Press, April 95).

[34] Brown and Sward, "Digital Downconversion Test Results witha Broadband L-Band GPS Receiver" 0-7803-2425-0/94 IEEE (NY: IEEE Press, 1994)
[35] Rolfes, M. "Field Trial Results of Superconducting Products in Wireless Communicaions", WESCON 96, p 109
36 Feher, K, Wireless Digital Communications (Prentice Hall, Upper Saddle River, NJ), 1995
[37] Fawcett, B. "Advancements in SRAM-Based FPGA Technology", WESCON 96, p 994
[38] Ono, *Introduction to Point Set Topology* (Baltimore, MD: Johns Hopkins University) 1974
[39] "Software drives multi-channel radio for digital battlefield" Military & Aerospace (Tulsa, OK: PennWall), Jun 97
[40] Bose et al, "Virtual Radio" IEEE JSAC on Software Radios, (NY: IEEE Press) 1998
41 Ziemer and Peterson, Digital Communications and Spread Spectrum Systems (New York: Macmillan, 1985)
[42] Recommendation H.320 Narrow-band visual telephone systems and terminal equipment (www.itu.int/publications/itu-t/ituth13.htm: International Telecommunications Union) 1998
[43] Coding of analogue signals by pulse code modulation (G.711 –G.712) and by methods other than PCM (G.720-G.729), (Geneva, Switzerland: International Telecommunications Union) 1998
[44] ietf references to internetworking
[45] Mouly & Pautet, "Evolution of the GSM System" IEEE PCS Magazine (NY: IEEE, Oct 1995).
[46] knowledge representation language
[47] Hennie, *Introduction to Computability* (Reading, MA: Addison-Wesley) 1997
[48] Simon and Szegedy "On the Complexity of RAM with various operation sets" 24th ANNUAL ACM STOC (ACM: VlCTORlA, B. C., CANADA) 1992
[49] Cutland, Computability , An Introduction to Recursive Function Theory (London: Cambridge University Press) 1980
[50] Machtey and Young, "An Introduction to the General Theory of Algorithms, (NY: North Holland)78
[51] xxx operational analysis throughput estimation
[52] xxx Hennie
[53] xxx machtey & young
[54] Nielson, H.R., and Nielson, F, "Higher-Order Concurrent Programs with Finite Communication Topology" (Extended Abstract) POPL 94- 1K14 (ACM; Portland Oregon, USA) 1994
[55] Milner, Tofte, and Harper, *The Definition of Standard ML*. (MIT Press: Boston, MA), 1990.
[56] Reppy, CML: "A Higher-order Concurrent Language" Proceedings of the ACM SIGPLAN '91 (Conference on Programming Language Design and Implementation (ACM Press) 1991.
[57] Moir, "Practical Implementations of Non-Blocking Synchronization Primitives", 1997 PODC 97, p219 (ACM: Santa Barbara CA) 1997.
[58] Herlihy and Shavit, "A Simple Constructive Computability Theorem for wait-free Computation", STOC 94, p243 (ACM: Montreal, Canada) 94
[59] Hoest and Shavit, "Towards a Topological Characterization of Asynchronous Complexity" (Preliminary Version), 1997 PODC, p199, (ACM: Santa Barbara, CA) 1997.
[60] Borowsky and Gafni, "A Simple Algorithmically Reasoned Characterization of Wait-free Computations (Extended Abstract)", 1h997 PODC, p188, (ACM: Santa Barbara CA) 97
[61] M. Fischer, N. Lynch, and M. Paterson, "Impossibility of Distributed Consensus with One Faulty Process," Journal of the ACM, vol. 32, no. 2, pp. 374–382, 1985.
[62] Luenberger's book on showing signal space as metric space
[63] Milewski, *The Topology Problem Solver*, (Piscataway, NJ: Research and Education Association) 1994
[64] laugenbacher complex adaptive systems
[65] Johnson, "Design and Control of Self-Organizing Complexes" Control Mechanisms for Complex Systems (Las Cruces, New Mexico: New Mexico State University) Dec 96
[66] h.320
[67] http://www.microsoft.com/ntserver/communications/tapi_wp.htm

[68] ITU, Specification and Description Language, Recommendations Z.100 (Geneva: ITU) 1991
[69] Methods for Testing and Specification (MTS); Strategy for the use of formal SDL for descriptive purposes in ETSI products, TR 101 081 V1.1.1 (Sophia Antipolis Cedex, FR: ETSI) 1997
[70] Mitola, J., *Adaptive Radio – Model Based Control of Soft Radios* (Kista, Sweden: KTH) 1998
[71] Upmal and Lackey, "SPEAKeasy, the Military Software Radio" IEEE Communications Magazine (NY: IEEE Press) 1995
[72] ITU, Open Distributed Processing, X.900-X.999 Recommendations (Geneva: ITU) Nov 95
[73] Eriksson and Penker, UML Toolkit (NY: John Wiley and Sons) 98
[74] MPI-2: Message Passing Interface Extensions (Knoxville, TN: University of Tennessee) 1998
[75] idl
[76] asn.1 citation
[77] Beyer, et al, *Radio Device API*, (Menlo Park, CA: SRI International) 1 Sep 97
[78] omg real time object

Appendix C Cognitive Radio: Making Software Radios More Personal[1]

Joseph Mitola III and Gerald Q. Maguire Jr.,

Royal Institute of Technology (KTH), Stockholm, Sweden.

Abstract: Software radios are emerging as platforms for multi-band multi-mode personal communications systems. Radio etiquette is the set of RF bands, air interfaces, protocols, spatial and temporal patterns that moderate the use of the radio spectrum. Cognitive radio extends the software radio with radio-domain model-based reasoning about such etiquettes. Cognitive radio enhances the flexibility of personal services through a Radio Knowledge Representation Language (RKRL). This language represents knowledge of radio etiquette, devices, software modules, propagation, networks, user needs, and application scenarios in a way that supports automated reasoning about the needs of the user. This empowers software radios to conduct expressive negotiations among peers about the use of the radio spectrum across fluents of space, time and user context. With RKRL, cognitive radio agents may actively manipulate the protocol stack to adapt known etiquettes to better satisfy the user's needs. This transforms radio nodes from blind executors of pre-defined protocols to radio-domain-aware intelligent agents that search out ways to deliver the services the user wants even if that user does not know how to obtain them. Software radio [1] provides an ideal platform for the realization of cognitive radio.

I. INTRODUCTION

A GSM radio's equalizer taps reflect the channel multipath structure. A network might want to ask a handset "How many distinguishable multipath components are you seeing?" Knowledge of the internal states of the equalizer could be useful because in some reception areas, there may be little or no multipath and 20 dB of extra Signal-to-Noise Ratio (SNR). Software radio processing capacity is wasted running a computationally intensive equalizer algorithm when no equalizer is necessary. That processing capacity could be diverted to better use, or part of the processor might be put to sleep, saving battery life. In addition, the radio and network could agree to put data bits in the superfluous embedded training sequence, enhancing the payload data rate accordingly[2].

Two problems arise. First, the network has no standard language with which to pose a question about equalizer taps. Second, the handset has the answer in the time-domain structure of its equalizer taps, but it cannot access this information. It has no computational description of its own structure. Thus, it does not "know what it knows." Standards-setting bodies have been gradually making such internal data available to networks through specific air interfaces, as the needs of the technology dictate. This labor-intensive process takes years to accomplish. RKRL, on the other hand, provides a standard language within which such unanticipated data exchanges can be defined dynamically. Why might the need for such unanticipated exchanges arise? Debugging new software-radio downloads might require access to internal software parameters. Creating personal services that differentiate one service provider from another might be

---

[1] This paper was published in the IEEE PCS Magazine's Special Issue on Software Radio, August, 1999

[2] This raises a host of questions about the control of such complex, adaptive agents, network stability, and the like.

enhanced if the provider does not have to expose new ideas to the competition in the standards-setting process. And the time to deploy those personalized services could be reduced.

Cognitive radio, through RKRL, knows that the natural language phrase "equalizer taps" refers to specific parameters of a tapped delay-line structure. This structure may be implemented in an Applications-Specific Integrated Circuit (ASIC), Field Programmable Gate Array (FPGA), or in an algorithm in a software radio. Since a cognitive radio has a model of its own internal structure, it can check the model to find out how the equalizer has been implemented. It then may retrieve the register values from the ASIC (e.g. using a JTAG port), or find the taps in the proper memory location of its software implementation. A radio that knows its own internal structure to this degree does not have to wait for a consortium, forum or standards body to define a "Level H33492.x7" radio as one that can access its equalizer taps. The network can pose such an unanticipated question in (a standard) RKRL and ***any*** RKRL-capable radio can answer it. To enable such a scenario, cognitive radio has an RKRL model of itself that includes the equalizer's structure and function as illustrated in Figure I-1.

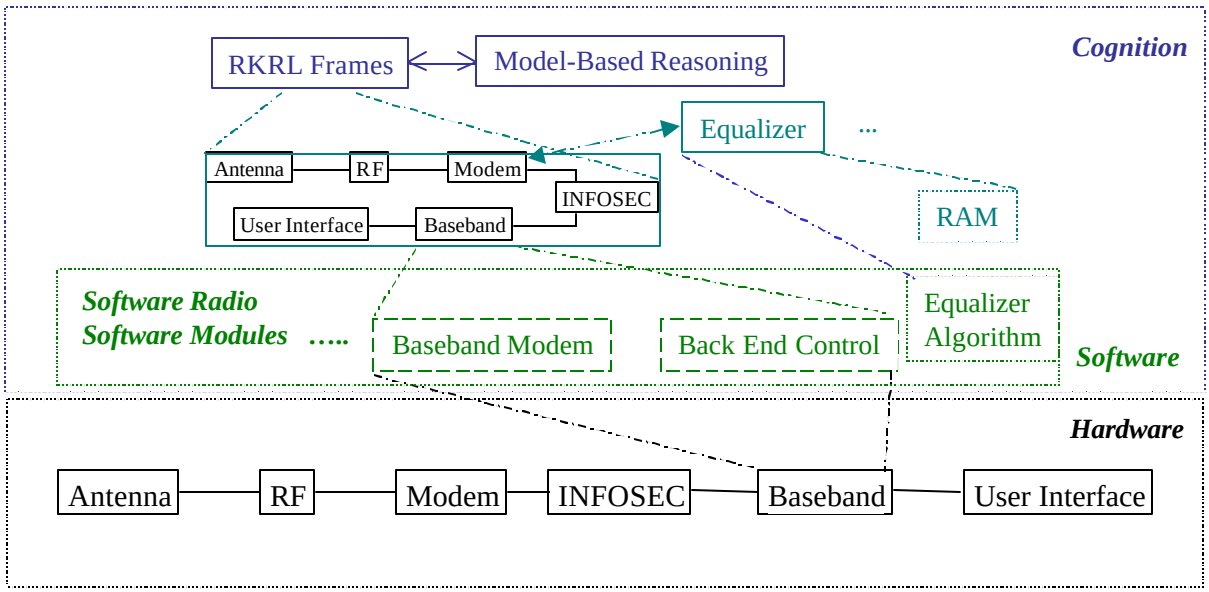


Figure I-1 Cognitive Radio

In this example, the radio hardware consists of the antenna, the RF conversion module, the modem and the other modules shown in the hardware part of the figure. The baseband processor includes a baseband modem and a back-end control protocol stack. In addition, this processor contains a cognition engine and a set of computational models. The models consist of RKRL frames that describe the radio itself, including the equalizer, in the context of a comprehensive ontology, also written in RKRL. Using this ontology, the radio can track the user's environment over time and space. Cognitive radio, then, matches its internal models to external observations to understand what it means to commute to and from work, to take a business trip to Europe, to go on vacation, etc.

Clearly, significant memory, computational resources, and communications bandwidth are needed for cognitive radio, so this technology may not be deployable for some time. In addition, a collection of cognitive radios may not require human intervention to develop their own

protocols. Initially, intervention will be required in order to assure that networks of such radios remain stable (or that we know who to blame if this is not the case). Networks of such radios are complex adaptive systems [2], the study of which is an emerging discipline concerned with the non-linear behavior of large collections of adaptive entities that have complex interactions. Although there are many technical challenges, the opportunities for enhanced personal services motivate the development of cognitive radio. This paper therefore outlines the key technical ideas behind cognitive radio, RKRL, and related research at KTH.

II. PERSONALIZED SERVICES SCENARIOS

The services enhancements to be enabled by cognitive radio are motivated by a set of use-cases [3] that require the radio to have an advanced degree of "understanding" of topics illustrated in Figure II-1. Next-generation PCS will know the location of handsets and wireless personal digital assistants (PDAs) to within 125 meters, for emergency location reporting. Location-aware research [4] is creating technologies for location-aware services, such as flexible directory services [5]. Cognitive radio adds locally-sensed recognition of common objects, events, and local RF context. Thus, for example, a cognitive radio can infer the radio-related implications of a request for a taxi to a specific address. It can then tell the network its plan to move from its present location to "Grev Turgatan 16." The network then knows that this user (with high probability) will move across three cell sites into a fourth within the next 10 minutes. If this user is headed for a conference center equipped with a local cell-phone jammer, it is unlikely to offer the usual load to the network after the taxi ride. Such exchanges could reduce uncertainty about the load offered to a network, potentially enhancing the efficiency of the use of radio resources.

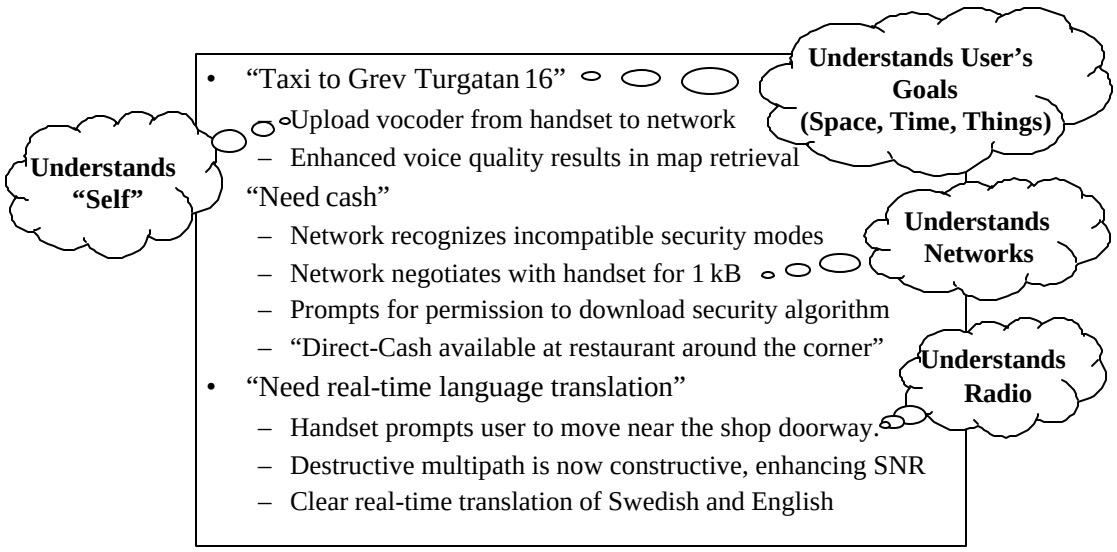


Figure II-1 More Personalized Services Concepts

Software radios as presently conceived cannot have such an intelligent conversation with a network because they have no model-based reasoning or planning capability and no language in which to express these things. For example, a software radio from the US may have the RF-access, memory and processing resources to operate in Sweden. If it lacks compatibility with "Release Level G" of the host service provider, then it will not work. A software radio cannot

"discuss" its internal structure with the network to discover that it can be re-configured to accept a download of the required software personality. Cognitive radio, however, employs a rich set of internal models useful for a wide range of such dialogs. In addition, the space-time models of the user, network, radio resources, and services personalize and enhance the consumer's experience. The analysis of such use cases yielded a large set of models, conceptual primitives and reasoning schema necessary for cognitive radio. What computer languages should be used to express these things?

III. RADIO-RELATED LANGUAGES

In addition to natural language, several computer-based languages are relevant to the expression of radio knowledge (Table 1). The International Telecommunication Union (ITU), for example, adopted the Specification and Description Language (SDL) in its Z.100 recommendations. SDL readily expresses radio state machines, message sequence charts, and related data dictionaries. The European Telecommunications Standards Institute recently adopted SDL as the normative expression of radio protocols, so one expects SDL modeling of radio to continue to expand. SDL, however, lacks primitives for general ontological knowledge needed, for example, to reason about a travel itinerary.

Table 1 Radio Knowledge Languages

Language	Strengths:	Lacks (or not designed for)
SDL [6]	State machines, message sequence charts, large user base, much encoded knowledge	Plan representation, uncertainty
UML [7]	General ontologies, structure, relationships	Hardware, RF propagation
IDL [8]	Interfaces, object encapsulation	General computing
HDLs [9]	Hardware, electronic devices, interfaces	Geospatial, plans
KQML [10]	General dialogs (ask/tell), tagged semantics	General computing
KIF [11]	Axiomatic treatment of sets, relations, numbers, frames; ontologies	General computing, Hardware, RF propagation

The Unified Modeling Language (UML) resulted from the unification of diverse object-oriented analysis, modeling, design, and delivery methods. This language readily expresses software objects, including attached procedures ("methods"), use cases, and the packaging of software for delivery. In principal, it can be used to model common-sense knowledge including plans, space, time, relationships, people – just about anything. In practice, it has a strong presence in software design and development, but it is weak in the modeling of hardware devices. In addition, although UML can provide a design framework for radio propagation modeling, the target languages are likely to be C or FORTRAN for computational efficiency in tracing tens of thousands of rays of radio waves.

The Common Object Request Broker Architecture (CORBA) defines an Interface Definition Language (IDL) as an implementation-independent syntax for describing object encapsulations. In addition to the 700 companies that comprise the Object Management Group, IDL is being

used by the Software-Defined Radio (SDR) Forum [12] to represent interfaces among the internal components of software-defined radios. Since this language is specifically designed to declare encapsulations, it lacks the computational power of general languages like C or Java. IDL excels at architecture integration (e.g. the interface to an equalizer ASIC), but not at expressing the functions and contributions of a component (e.g. the enhancement of Bit Error Rate at low SNR).

The hardware description languages (HDLs), primarily Verilog HDL and VHDL readily express the internal structure of ASICs, and the personalities of FPGAs. But cognitive radio does not need the level of detail present in most HDL data sets. Moreover, it needs to know the functions and contributions so that it can make tradeoffs, create plans, and reprogram itself. While the documentation package associated with HDL may provide some of this insight, the information is not in a computationally accessible form.

The Knowledge Query and Manipulation Language (KQML), on the other hand, was explicitly designed to facilitate the exchange of such knowledge. Based on "performatives" such as "tell" and "ask", KQML readily express the dialog about "equalizer taps" and "multipath" by introducing a few new "tags". The KQML plan to take a taxi from the information kiosk to Grev Turgatan 16 uses the Tell performative to tell the network of the plan as shown in Figure III-1. In this example, the radio also warns the network that its user is composing some email and so will need either a DECT data channel or the GSM packet radio service (GPRS) while in transit.

```
(Tell :language RKRL :ontology Stockholm/Europe/Global/Universe/Version 0.1
:Move_Plan (:owner User (:from Kiosk :to "Grev Turgatan 16") :distance 3522 m
(:via (Taxi :probability .9) (Foot :probability 0.03))
(:PCS-needs (:DECT 32kbps) (:GSM GPRS) (:backlog Composing-email)))
```

Figure III-1 KQML Expression of a Plan

The ontology performative could invoke an existing ontology or could express the local context, as in this case. It normally would be defaulted unless it changed. The other declarations are self evident, which is one of the strengths of KQML. Like IDL, however, KQML is an interface language. Although, for example, one can express rules from a knowledge base using KQML, one must translate these rules into a convenient internal form (e.g. LISP or PROLOG) in order to use them. In addition, the expression of general spatial knowledge, such as the three-dimensional structure of adjacent city blocks, is better expressed in structured arrays than in KQML. KQML could be used to send changes to such arrays, however.

The Knowledge Interchange Format (KIF) provides an axiomatic framework for general knowledge including sets, relations, time-dependent quantities, units, simple geometry, and other domain-independent concepts. Its main contribution is strong axiomatization. It has a LISP-like structure, and, like IDL and KQML is not specifically designed for "internal" use, like C or Java.

Finally, most radio knowledge is represented in natural language. It lacks precision, but in some sense has the ultimate in expressive power, particularly if one includes graphics and multimedia as natural language. Natural language suffers from ambiguities and complexity that at present

limit its use as a formal language. RKRL Version 0.1 was created to fill the voids in the expressive power of the computer languages while enforcing a modicum of structure on the use of natural language.

IV. Cognitive Radio As A Chess Game

RKRL is supposed to represent the domain of information services that use software radios for mobile connectivity. Since a software radio has a choice of RF bands, air interfaces, data protocols, and prices to be paid, in competition with other users, the domain is analogous to a chess game. The network may orchestrate the game, or in some bands (e.g. the US Instrumentation, Scientific, and Medical bands), the cognitive radios will simply compete with each other, hopefully with some radio etiquette. The game board is the radio spectrum with a variety of RF bands, air interfaces, smart antenna patterns, temporal patterns and spatial locations of infrastructure and mobiles. RKRL must provide a consistent way of describing this game board. The future wireless PDAs are the game pieces. What moves are legal? How will one move impact others in the neighborhood? If a game piece expresses its future needs or plan for use of services to the network, can the network better orchestrate the use of radio resources? And to what degree should the way in which spectrum is used over physical space, code space, power, parameter space, and time be defined by the mobile units themselves? No one knows the answers to these questions because software radios are just emerging. But the need to address them in the future seems clear. RKRL should express the game board and the legal moves.

Davis [13] defines micro-worlds as performance domains for naïve physics. Cognitive radio consists of the multiple micro-worlds (the meso-world) represented as in Figure IV-1.

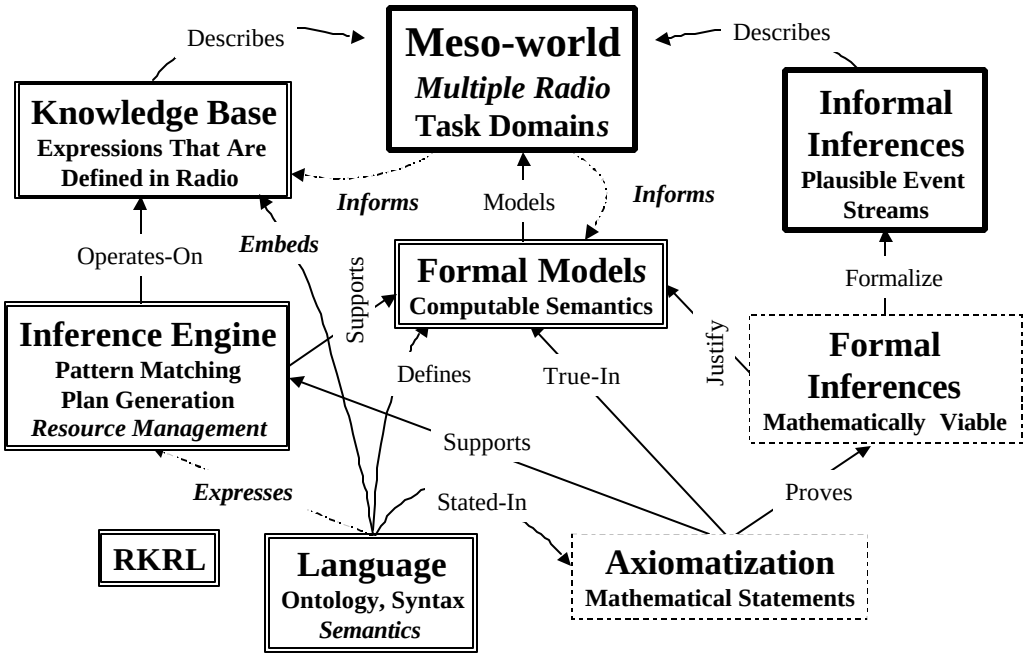


Figure IV-1 Meso-world structure of Cognitive Radio

The meso-world of RKRL Version 0.1 consists of the forty-one micro-worlds summarized in Figure IV-2. Each is structured according to formal models, and each is described in a knowledge base. Competence comes from the pattern matching and plan generation capabilities of a cognition cycle, mediated by the related inference engines. RKRL includes syntax and ontological information. Parsing an RKRL statement includes interpreting that statement in terms of the RKRL radio ontology and knowledge base. Thus, words, including KQML tags, have a meaning that is fixed in a given context (although a single word can have different meanings in different contexts). Thus, the scope of RKRL includes the formal models, the knowledge base, the inference engine, multiple syntaxes, and a radio ontology.

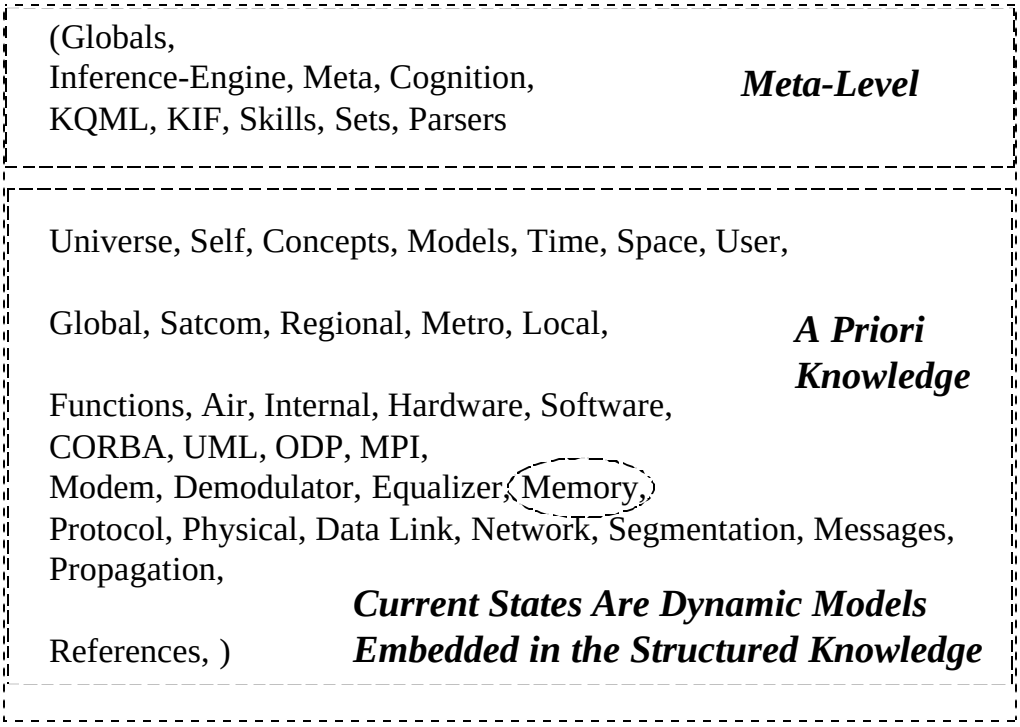


Figure IV-2 RKRL Micro-worlds

The expression of syntax in RKRL permits one to embed knowledge from external representations (especially SDL and UML). RKRL may now be described syntactically.

V. RKRL

Instead of attempting to replace SDL, UML, IDL, or KQML, RKRL integrates them through model-based reasoning as follows. RKRL is a parallel frame language. Each RKRL statement is a frame:

<frame> = [<Handle>, <Model>, <Body>, <Context>].

The frame expresses a relationship between the handle and body, in a given context. The <model> part defines the exact relationship being expressed. Handles should be thought of as names for things. If a thing contains other things, it can be viewed as an object. If not, then it is a terminal constant. Frames are interpreted in parallel, like the cells in a spreadsheet. For

example, the following RKRL statement says that South America is part of the Global Plane in the Physical World Model of the Universe of RKRL Version 0.1. Additional frames assert Europe, etc. into the Global Plane. Since the Global plane is part of the Universe, it can be thought of as an attribute of Universe, with a value that is a set of regions of the world. But it can also be thought of as just a list of the names of countries. These semantics (e.g. object, list, etc.) are *NOT* part of the semantics of RKRL. Instead, the semantics are explicitly declared and coded in RKRL using formal computational models.

Handle	***Model***	***Body***	***Context***
Global Plane	Contains	South America	Physical World Model/ Universe/ RKRL/ Version 0.1

Figure V-1 An RKRL Frame Asserting that The Global Plane Contains South America

For example, the word "Contains" is a verb from natural language, used in an obvious way. But Contains is also a formal model defined in the Models micro-world using the following statements.

Handle	***Model***	***Body***	***Context***
Contains	<Process>	SetAccumulate	Models/Universe/RKRL/Version 0.1
SetAccumulate	Excel	<Control><Shift>S	Contains/ Models/Universe/RKRL/Version 0.1
Contains	<Test>	modelVar = "Contains"	Models/Universe/RKRL/Version 0.1
Contains	<Domain>	Sets	Models/Universe/RKRL/Version 0.1
Contains	<Range>	Sets	Models/Universe/RKRL/Version 0.1
Contains	Definition	This string-model asserts that the body is a member of the handle.	Models/Universe/RKRL/Version 0.1

Figure V-2 Statements Defining the Verb "Contains" As A Formal Model

Contains, the model, is defined in terms of a <domain> (the micro-world of Sets), a <range>, a <test> for membership, and a <process> for finding members in a local context. The first frame says that "Contains" has a <process> called SetAccumulate. The fourth line says that Contains is defined over Sets (which is a complete micro-world). The items in <brackets> are defined in the Meta micro-world. The <test> process consists of a chunk of Excel Visual Basic for Applications (VBA) macro language that will test a frame with a standard binding (modelVar is bound to the Model part of the frame). These string models tell the RKRL interpreter (embryonic at present) how to use chunks of code to construct programs that create and manipulate objects, perform model-based reasoning, and otherwise control the software radio platform. To attach an existing VBA macro to an entity, one states (as in line 2 above) that its Excel model is <Control><Shift>S, the associated keyboard macro. In addition, one may re-define contains in some other context. If it is undefined in a local context, the interpreter searches the micro-worlds ascending the context hierarchy, and then looks across micro-worlds

until it either finds a definition or sets a Goal of "Getting" a definition. In RKRL to Get includes to Create (through inferencing) and to Inquire (e.g. of the user or of the network).

A. Knowledge About Equalizers

For a more radio-oriented example, consider the adaptive equalizer. RKRL knows about the equalizer running on its own platform from the following context: Equalizer/ Demodulator/ Modem/ Internal/ Fine Scale/ Immediate/ Local/ Metropolitan/ Regional Plane/ Global Plane/ Physical World Model/ Universe/ RKRL/ Version 0.1. It also knows about a generic equalizer in Equalizer/Demodulator/Modem/Concepts/RKRL/Version 0.1. Facts that are known a-priori of an equalizer include a frame [Equalizer, Property, Reduces inter-symbol interference …]. The natural language phrase "reduces inter-symbol interference" is parsed because the Concept of reduce has the Excel fragment "handleVar(bodyVar)<bodyVar" and ISI is defined in the physical layer as "inter-symbol interference" with the property that it increases BER. Since all the elements reduce to either Excel, VBA, or a call to an embedded model, the frame is completely interpretable.

The equalizer is defined from the <domain> IF-signal onto the <range> IF-Signal. Its taps are defined using statements like: [Tap-0, 1.2745, Numerical model, Taps/Delay Line/ Equalizer/…]. The output is defined in a frame [Weighted-result, *, Numerical model, …], where * is the Excel expression:

> =Weight 3 Model * Tap 3 Model + Weight 2 Model * Tap 2 Model + Weight 1 Model * Tap 1 Model

If the present RKRL were embedded in a PDA, the a-priori model of a three-tap equalizer would be as above, but the Internal model would be a Dynamic model. Dynamic models contain the values from the current system that they are modeling. A Unix Stream can be a dynamic model, for example. Thus, cognitive radio could tell the network about its equalizer by binding its generic model to the Dynamic model stream and reporting the results in KQML to the network. Since such values change as a function of time, RKRL will access (and log) signals as fluents [14] in order to detect regular patterns.

The current version of RKRL is implemented in Visual Basic attached to 41 Excel spreadsheets. Object linking and embedding from Excel allows RKRL to access almost any existing software as an executable model. Thus, instead of writing the large number of subordinate models needed for a comprehensive RKRL, the RKRL framework points to those that exist. In addition, KQML, SDL, IDL, and UML primitives are represented in RKRL. One of the benefits of this approach is an ability to express a given item in more than one "standard" way. Another benefit is the ability to parse expressions from other languages in order to extract existing knowledge for use in cognitive radio.

B. Spatial Inference Hierarchy

RKRL embeds a standard spatial inference hierarchy for space and time as shown in Table 2. Each of the planes consists of objects with associated space-time properties. RKRL also declares ways in which the radio can autonomously obtain information about objects on that level. The global plane, for example, divides the Earth into large regions. The properties of the global plane change in annual cycles, e.g., through annual holiday patterns of travel. RKRL statements at this level that define the components of standard annual cycles including seasons of the year, weather, and holidays. The radio can get information about its user's interaction with the global plane by examining the user's travel itinerary. Other planes contain objects appropriate to that level of abstraction, including space-time characteristics and information sources.

The lowest level of this hierarchy represents the physical architecture of the software radio. It describes antennas, digital signal processors, memory (RAM and ROM), user interface devices, etc. in terms of physical capabilities and interconnections. Although there is nothing to preclude RKRL from invoking a complete HDL description of the radio, goals of cognitive radio concern inference about higher level aspects of radio etiquettes. RKRL micro-worlds in the Internal plane embed the architecture framework, applications programmer interface (API), and IDL of the SDR Forum.

Table 2 Physical World Inference Hierarchy

	Plane	**Objects**	**Space**	**Time**	**Information Sources**
1	Global	Regions	10,000 km	1 yr	Travel Itinerary
2	Regional	Cities	1000 km	1 week	Weekly Planner
3	Metropolitan	Districts	100 km	1 day	Commuting Pattern
4	Local	Buildings	1 km	1 hr	GPS, Lunch?
5	Immediate	Rooms	100m	1sec-1min	Dead reckoning
6	Fine Scale	Body Parts	1m	1 usec	Equalizer Taps
7	Internal (Radio)	HW, SW	.1m	1 nsec	Architecture

C. Model-Matching

Detailed models of radio functions are embedded in RKRL 0.1 for each micro-world in which competence is required. The following, for example, is an executable model of the segmentation of a message into packets (from the Segmentation micro-world).

Parameter	**Value**
Message	SAMPLE MESSAGE
Payload Size	40
Payload Bytes	5
Packet Number	1
Payload	E MES

Figure V-3 Excel Model of A Message Being Parsed Into Packet Payloads

If a cognitive radio sets the packet number to 0, the payload becomes SAMPL, the first 5 octets of the outgoing message. To determine the impact of changing the protocol, the system first copies the values of the model to temporary RKRL frames. It then changes the parameters to correspond to its hypothesized protocol (e.g. to a 7 byte payload). Finally, it compares the values of the model to the previously stored values to determine how the change of protocol will change the payload. It logs this to a new RKRL frame as a cause-and-effect relationship.

VI. The Cognition Cycle

RKRL supports the cognition cycle illustrated in Figure VI-1

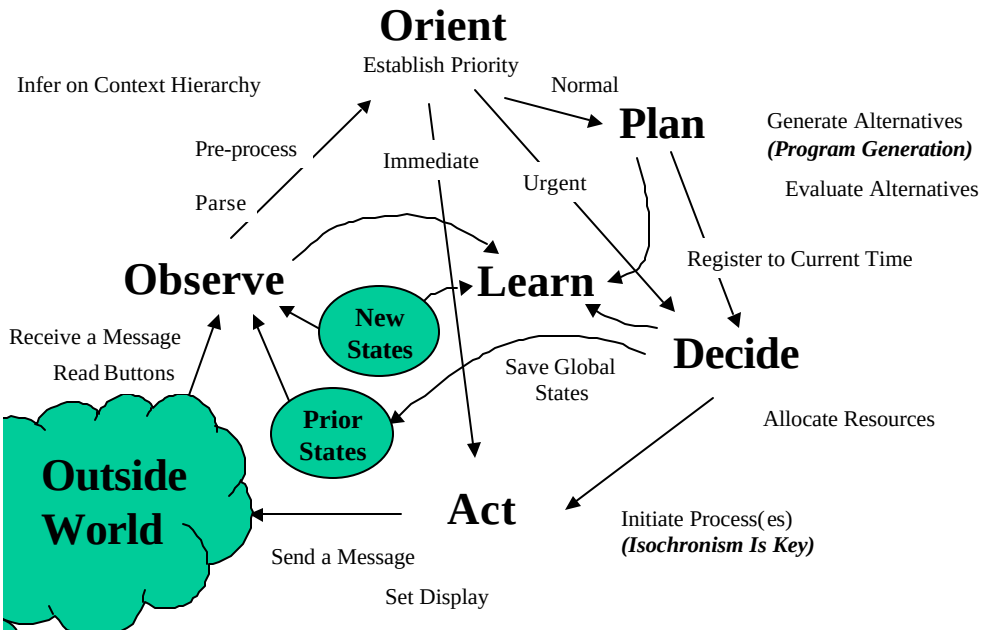


Figure VI-1 The Cognition Cycle

The outside world provides stimuli. Cognitive radio parses these stimuli to extract the available contextual cues necessary to performance of its assigned tasks. It might analyze GPS coordinates plus light and temperature to determine whether it is inside or outside of a building. This type of processing occurs in the Observe stage of the cognition cycle. Incoming and outgoing messages are parsed for content, including the content supplied to/by the user. This yields contextual cues necessary to infer the urgency of the communications and related internal tasks. This task is akin to topic spotting in natural language processing. Even relatively high word error rates can result in high probability of detection and low false alarm rate in detecting ordinary events. Thus, the radio "knows" it is going for a taxi ride (with some probability) if the user packets at the wireless information kiosk order a taxi. If the main battery has just been removed, however, the Orient stage immediately Acts to save data necessary for a graceful start-up and to shut the system down. Loss of carrier on all available links (e.g. because of entering a building) can result in urgent steps to restore connectivity, such as scanning for an in-building PCS or RF LAN. Most other normal events might not require such time-sensitive responses, resulting in the Plan-Decide-Act cycle. The Act step consists of allocating computational and

radio resources to subordinate (conventional radio) software and initiating tasks for specified amounts of time. RKRL also includes some forms of supervised and unsupervised learning.

VII. Conclusion

Software radios provide a vast untapped potential to personalize services. But the contemporary process of modifying radio etiquettes is extremely labor-intensive. In part, this is because there is no generally accepted way of representing radio knowledge. This limits the flexibility and responsiveness of the radio to the network and to the user. RKRL may provide some insights into how to better automate this process. Cognitive radio, built on RKRL, is envisioned as a competence system over the domain of radio resources and protocols. Its agent knowledge and inference mechanisms are under development, as is the initial "critical-mass" definition of RKRL.

Finally, RKRL is designed to be used by software agents that have such a high level of competence, driven in part by a large store of a-priori knowledge, that they may accurately be called "cognitive". This goal may be very far off, or it may emerge from the current research program. Cognitive radio approaches the software radio as a micro-world. But radio engineering is such a large, complex world that will take a lot of effort to describe it in computationally accessible, useful ways. The present research is therefore offered as a mere baby-step in a potentially interesting research direction.

---

[1] Mitola, J., *Software Radio: Wireless Architecture for the 21st Century* (103255.2507@compuserve.com: Mitola's STATIS*faction* ISBN 0-9671233-0-5)

[2] Kaufman, S., *At Home In The Universe* (NY: Oxford University Press) 1995

[3] Mitola, J., *Cognitive Radio*, Licentiate Proposal (Stockholm, KTH) December, 1998

[4] Location-Aware Computing and Communications Home Pages, KTH, Sweden and Wollongong University, Australia (www.elec.uow.edu.au/people/staff/beadle/badge/location_aware.html).

[5] Henning Mass, "Location Aware Mobile Applications Based On Directory Services", Mobicom 97 (www.acm.org: Association of Computing Machinery) 1997

[6] ITU, Specification and Description Language, Recommendations Z.100 (Geneva: ITU) 1991

[7] Eriksson and Penker, *UML Toolkit* (NY: John Wiley & Sons, Inc.) 1998

[8] Mowbray and Zehavi, *The Essential CORBA*, Object Management Group (NY: Wiley) 1995

[9] Object GEODE (Paris, FR: Verilog) 1998

[10] Finin, T., "KQML -- A Language and Protocol for Knowledge and Information Exchange,: *International Conference on Building and Sharing of Very Large-Scale Knowledge Bases*, Tokyo, December, 1993

[11] Genesereth and Fikes, "Knowledge Interchange Format Version 3.0 Reference Manual" Logic Group Report Logic 92-1 (Palo Alto, CA: Stanford University) 1992

[12] SDR Forum Web Site www.sdrforum.org

[13] Davis, E., "The Naïve Physics Perplex", *AI Magazine* (Menlo Park, CA: AAAI) Winter 1998

[14] Cohen, et al., "Neo: Learning Conceptual Knowledge by Sensorimotor Interaction with an Environment" Proc Autonomous Agents 97 (www.acm.org: Association of Computing Machinery) 1997

Appendix D Cognitive Radio for Flexible Mobile Multimedia Communications [1]

Joseph Mitola III

Royal Institute of Technology (KTH), Stockholm, Sweden.

Abstract: Wireless multimedia applications require significant bandwidth, some of which will be provided by third-generation (3G) services. Even with substantial investment in 3G infrastructure, the radio spectrum allocated to 3G will be limited. Cognitive radio offers a mechanism for the flexible pooling of radio spectrum using a new class of protocols called formal radio etiquettes. This approach could expand the bandwidth available for conventional uses (e.g. police, fire and rescue) and extend the spatial coverage of 3G in a novel way. Cognitive radio is a particular extension of software radio that employs model-based reasoning about users, multimedia content, and communications context. This paper characterizes the potential contributions of cognitive radio to spectrum pooling and outlines an initial framework for formal radio-etiquette protocols.

Keywords: Software radio, cognitive radio, spectrum management, software agents.

I. BACKGROUND

A. Software Radio and SDR

A software radio [1] is a multi-band radio capable of supporting multiple air interfaces and protocols through the use of wideband antennas, RF conversion, Analog to Digital Converters (ADCs) and DACs. In an ideal software radio, all the aspects of the radio (including the physical air interface) are defined in software on general-purpose processors. For some air interfaces such as Wideband Code Division Multiple Access (W-CDMA), such an ideal implementation may not be practical for one reason or another (e.g. power consumption). As processor technology advances, however, air interfaces that require Application Specific Integrated Circuits (ASICs) today may be implemented on general-purpose processors. The software-defined radio (SDR) therefore compromises the software radio ideal in order to implement practical high-performance devices and infrastructure with current technology. SDRs are implemented using an appropriate mix of ASICs, Field Programmable Gate Arrays (FPGAs), Digital Signal Processors (DSPs) and general-purpose microprocessors. The architecture and computational aspects of the ideal software radio have been defined formally [2]. In addition, the global SDR forum has defined an architecture framework, object models and other recommendations for SDR [3].

B. Cognitive Radio

Cognitive radio signifies a radio that employs model-based reasoning to achieve a specified level of competence in radio-related domains [4]. Cognitive radio architectures being

[1] Best Paper of the 6th International Workshop on Mobile Multimedia Communications, Nov 99

investigated at KTH employ the cognition cycle illustrated in Figure I-1. The outside world provides stimuli. Cognitive radio parses these stimuli to recognize the context of its communications tasks. Incoming and outgoing multimedia content is parsed for the contextual cues necessary to infer the communications context (e.g. urgency). Thus, for example, the radio may infer that it is going for a taxi ride (with some probability) if the user ordered a taxi by voice and is located in a foreign country. The Orient-stage decides on the urgency of the communications in part from these cues in order to reduce the burden on the user. Normally, the Plan-stage generates and evaluates alternatives, including expressing plans to peers and/or the network to obtain advice. The Decide-stage allocates computational and radio resources to subordinate (conventional radio) software. The Act-stage initiates tasks with specified resources for specified amounts of time.

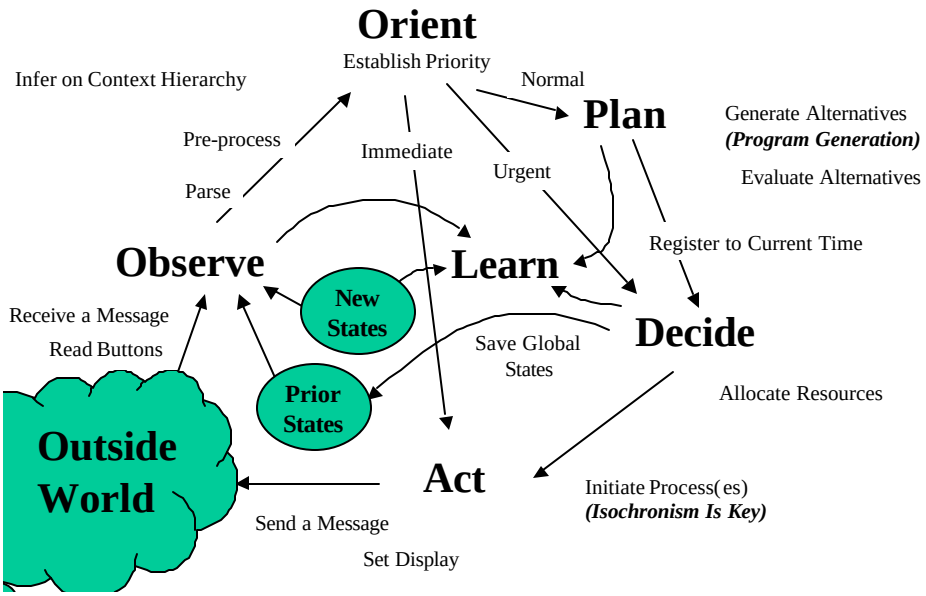


Figure I-1 The Cognition Cycle

If the main battery has just been removed, however, the Orient stage would immediately invoke the Act-stage to save data necessary for a graceful start-up after shutdown. Unexpected loss of carrier (e.g. because of departing a non-intelligent building) of an RF LAN that is in use can result in an urgent Decide-stage to restore traffic flow, such as via a more expensive 3G network. Most other normal events might not require such time-sensitive responses, resulting in the Plan-Decide-Act cycle of the figure. Cognitive radio also includes some forms of supervised and unsupervised machine learning.

Cognitive radio is a goal-driven framework in which the radio autonomously observes the radio environment, infers context, assesses alternatives, generates plans, supervises multimedia services, and learns from its mistakes. This observe-think-act cycle is radically different from today's handsets that either blast out on the frequency set by the user, or blindly take instructions from the network. Cognitive radio technology thus empowers radios to observe more flexible radio etiquettes than was possible in the past.

C. SDR-Enabled Infrastructure and Mobile Multimedia Devices

Recently Mitsubishi and AT&T announced the first "four-mode handset." The T250 can operate in TDMA mode on 850 or 1900 MHz, in first generation Analog Mobile Phone System (AMPS) mode on 850 MHz, and in Cellular Digital Packet Data (CDPD) mode. This is just the beginning of the multiband, multimode, multimedia (M^3) wireless explosion. In the not-too-distant future, software-radio based Personal Digital Assistants (PDAs) could access a satellite mobile services, cordless telephone, RF LAN, GSM, and 3G W-CDMA. Such a device could affordably operate in octave bands from .4 to .96 GHz, (skip the air navigation and GPS band from .96 to 1.2 GHz),1.3 to 2.5 GHz, and from 2.5 to 5.9 GHz. Octave bands enhance antenna efficiency and reduce the cost of wideband RF conversion components. Not counting satellite mobile and radio navigation bands, such radios would have access to over 30 mobile sub-bands in 1463 MHz of potentially *sharable* outdoor mobile spectrum. The upper band provides another 1.07 GHz of *sharable*[2] indoor and RF LAN spectrum. This wideband[3] radio technology will be affordable first for infrastructure, next for mobile vehicular radios and later for handsets and PDAs. Such software radio technology expands opportunities for the dynamic sharing of spectrum. But it is the well-heeled conformance to the radio etiquettes afforded by cognitive radio that could make such sharing practical.

II. POOLED RADIO SPECTRUM

A. Pooling Strategy

Present commercial wireless architectures are network-centric and constrained by spectrum allocations. Nick Negroponte [5] pointed the way with his spectrum management algorithm of the future: "If it moves, give it spectrum; if it doesn't, give it fiber." A slightly more practicable strategy accessible through SDR and cognitive radio would be to pool mobile spectrum:

"If it moves, give it spectrum pool precedence; if it doesn't, make it pay."

Satellites and aircraft move rapidly and/or cover large areas, so the bands dedicated to these vehicles would not be pooled. Broadcast television stations and 2 GHz microwave, however, would have to pay for the privilege of using prime spectrum in the middle of the mobile bands. The cost would be near zero in rural areas where the radio spectrum is uncrowded, so low cost broadcast and connectivity to rural areas would not be negatively affected. The cost would be appropriately larger in the densest urban areas where there is simply not enough suitable spectrum to economically meet the demand for mobile wireless. In addition, in those areas, the proliferation of fiber and cable makes such broadcast less necessary than in rural areas.

Conversely, however, those who currently "own" the spectrum could charge rent by the second, minute, or hour, as dictated by the marketplace. Federal, state, and local governments could generate revenue streams by literally renting channels that are not currently in use, and for which there is no need for some agreed-upon time interval (e.g. the next 10 minutes). The radio

---

[2] Although 5.9-2.5 = 3.4 GHz, only 1.07 GHz of this spectrum is allocated to mobile - hence *sharable* - subbands.

[3] Radios operating at 2.5 GHz are not expensive. Efficient wideband operation from 2.5 to 5.9 GHz, however, *is* expensive today. Costs are dropping as implementation technologies continue to advance.

etiquette would specify that the cognitive radios would change bands and modes when legacy[4] radios enter the bands. Thus, police forces would not have to procure new mobile radios. But, every police station, fire house, military facility, and taxi company could readily become an M^3 cell site. These new base stations could pay for the SDR cell-site electronics with revenues from renting unused channels to 3G service providers during peak hours[5]. Again, the radio etiquette can specify that emergency vehicles take precedence over commercial rental traffic. The cognitive radios using the bands would gracefully defer as they monitor the rented channels for FM radios, Tetra [6] users, and any other authorized legacy emissions. Since a variety of subbands and modulation schemes are in use by the public service community, the reliable identification of authorized users is sometimes not an easy task.

Jens Zander has pointed out that the shortage of spectrum can be viewed instead as a shortage of affordable cell sites [7]. Spectrum pooling both increases the number of sites and integrates the multi-mode spectrum so that the quantities needed for multimedia wireless applications are more affordable. The spectrum rentals (using e-cash or bartered spectrum channel-seconds, s-cash) would happen so fast that they must be accomplished by computer. Cognitive radios could use their knowledge of the RF environment, multimedia content, and communications context (e.g. a life-threatening emergency) to barter according to guidelines specified by the spectrum managers and represented in the radio etiquette protocol.

Clearly, managers set the criteria and make long-term spectrum leases. But the cognitive radios would rent out the short-term locally available spectrum that is not instantaneously in use, establishing spot prices as a function of time, bandwidth, interference levels, radiated power, location, and perhaps other parameters. Cognitive radios could rent spectrum for a second (e.g. to upload a brief email message), a minute (e.g. for part of a voice call), an hour (e.g. for a video teleconference) or more. Spectrum management authorities would establish the general etiquettes and constraints, but the market would set the price. Cognitive radios would use their spectrum awareness and goal-driven behavior to mutually assure conformance to etiquette, and to identify and report offenders to human authorities. Such a pooled spectrum strategy could accomplish dynamically with distributed control technology what could not be contemplated with today's centralized allocation-based control – the cost-effective efficient pooling of formerly scarce mobile radio spectrum for fair and equitable use when and where needed.

The resulting spectrum management process, however, would be akin to a chess game where the board is the radio spectrum, the players are the cognitive radios and networks, and the winners are the users. To see how the game might be played, the next section examines the "chess board" – the potentially pooled mobile radio spectrum. Subsequent sections describe the "legal moves" – specified by the radio etiquette protocol needed to maintain equity and order.

---

[4] A legacy radio is an existing non-cognitive radio operating in an air interface mode assigned to that band.

[5] The resulting relative glut of spectrum could drive costs down in the short term, enhancing profit margins to provide the capital necessary to build out today's hardware-constrained infrastructure to M^3-capable SDR infrastructure.

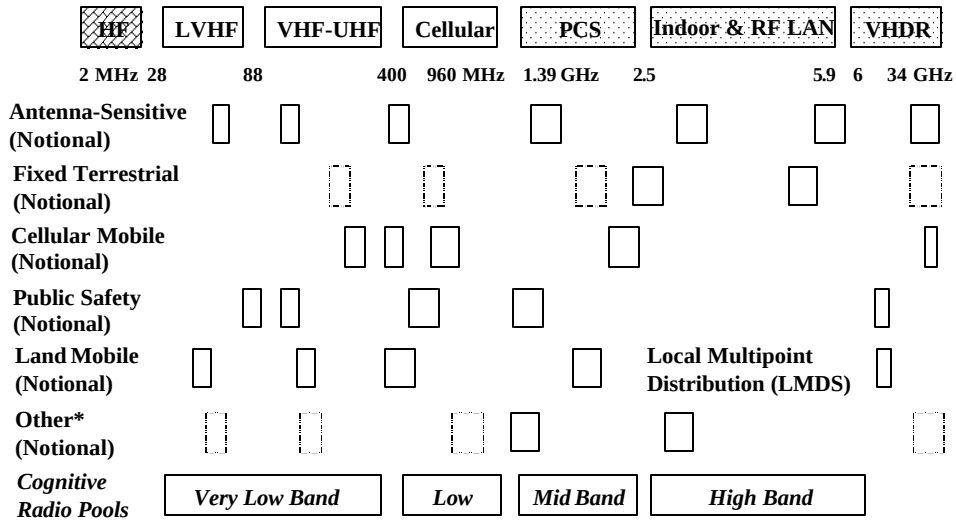
B. Pooled Mobile Spectrum Parameters

The laws of physics impose limits on the spectrum that is useful for pooled terrestrial mobile-multimedia applications. The HF bands and below, for example, have limited bandwidth (practically speaking, a few 10's of kHz, although heroic experiments have achieved megabit per second data rates in truly unique circumstances). HF also propagates for thousands of miles, a range that exacerbates cochannel interference. Bands above 6 GHz rely on directional antennas for reasonable data rates at reasonable ranges. HF and upper SHF are therefore not suitable.

Table II-1 Mobile Spectrum Pools

Band	**RF_{min} (MHz)**	**RF_{max} (MHz)**	**W_c**	**Remarks**
Very Low	26.9	399.9	315.21	Long range vehicular traffic
Low	404	960	533.5	Cellular
Mid	1390	2483	930	PCS
High	2483	5900	1068.5	Indoor and RF LANs

The four spectrum pools of Table II-1, however, are ideally suited to mobile applications. The *very low band* of this mobile spectrum regime penetrates buildings and propagates well in rugged terrain. The *low band* has the best propagation for high-speed terrestrial mobile traffic, in part, because auto and rail traffic is supported with relatively low infrastructure density. *The mid band* is best for Personal Communications Services (PCS) with its higher infrastructure density. In addition, the *high band* has the large coherent bandwidth for high data rate Internet and mobile video teleconference applications. 3G waveforms could be used in any of these bands, but are best suited for the *low* and *mid bands*. W_c is the total spectrum that could participate in the spectrum pool based on an analysis of US, Canadian, and UK spectrum allocations [4]. W_c does not include satellite, aircraft, radio navigation, astronomy, or amateur bands, which are not suited for pooling. The pooling concept is illustrated in Figure II-1.



***** Includes broadcast, TV, telemetry, Amateur, ISM; VHDR = Very High Data Rate

Figure II-1 Fixed Allocations vs. Pooling with Cognitive Radio Etiquette

A cognitive radio-access network could evolve the operating parameters shown in Table II-2. For simplicity, the entire population offers load (100% penetration). With pooling, each mobile outdoor user would have an average of 432 kbps. This assumes today's infrastructure density and 2G-equivalent bandwidth efficiency (0.2 Mbps/MHz/cell). These rates are gross data rates not discounted for Quality of Service (QoS), which can lower these rates substantially if low bit error rates are required (e.g. for file transfers) [8]. They also do not include signaling overhead. On the other hand, 3G technology is supposed to achieve 0.45 Mbps/MHz/cell, so the rates are representative of the range of rates achievable with a mix of 2G and 3G technology.

Table II-2 Illustrative Cognitively Pooled Radio Access Network Parameters

Parameter	**Illustrative Range of Values**	**Remarks**
Total Spectrum	0.4 to 2.5 GHz	1.463 GHz pooled
Duplexing	Frequency Domain (FDD)	Evolved from cellular services
Voice Channel	8 1/3 kHz-equivalent, TDMA or CDMA	Evolved from second generation
Channels per cell	25088	Usable, including 6:1 reuse and FDD
Coverage area	4000 square kilometers	The size of Washington, DC
Number of cells	40 commercial (plus 40 public sites*)	5.5 km average cell radius (3.9 km)
Population	609,000	The entire population of Washington, DC
Offered demand	0.1 Erlang	Multimedia level (vs. 0.02 for voice user)
Demand per cell	1522 Erlangs	Drops to 761 considering public cell sites
Spectrum per user	160.7 kHz (320.4 kHz with public sites)	.22 to .64 Mbps/ user (.4 to 1.28 Mbps)

* Public sites are towers of police, fire, military, and other government/ public facilities pooling spectrum

With spectrum pooling, then, multimedia bandwidths can be achieved without a major increase in the number of cell sites. In part, the participation of public facilities increases the number of sites. In addition, the pooling of spectrum is more efficient than block allocations. Through cognitive radio etiquette, police, fire, and rescue units participating in spectrum pooling will have precedence for spectrum. They can also communicate seamlessly via the shared SDR cell sites.

C. Multimedia Implications

Pooled spectrum is attractive for wireless multimedia applications for the reasons listed in Table II-3.

Table II-3 Multimedia Implications of Pooled Spectrum

Feature	**Implications**
Sharing of Narrowband Channels	Reduces costs of new uplink channels for Internet browsing and other multimedia services
Spectrum Block Rentals (e.g. 1-5 MHz)	New spectrum for multimedia downlinks
Increased Infrastructure Density	Greater availability of affordable multimedia
Short Term Spectrum Rentals	Accommodate peak demand; Introduce new services incrementally

The benefits seem attractive. What about guarantees and fairness of use? In order for the approach to work, all emergency services, government functions, private, and commercial users will participate actively or passively a cognitive radio etiquette protocol that makes the difference between new levels of spectrum efficiency and chaos.

III. COGNITIVE RADIO ETIQUETTE

Radio etiquette is the set of RF bands, air interfaces, protocols, spatial and temporal patterns, and high level rules of interaction that moderate the use of the radio spectrum. Etiquette for spectrum pooling includes the spectrum renting process, assured backoff to authorized legacy radios, assured conformance to precedence criteria, an order-wire network, and related topics.

A. Renting Spectrum

An initial protocol framework for renting radio spectrum is illustrated in Figure III-1. The time line shows the power levels in the rented channel, differentiating signals of renter and owner. The offeror initiates the process by posting an "Advertise" flag in the channel that is for rent. This in-band signaling accomplishes multiple goals. First, it unambiguously identifies the frequency, bandwidth (through its spectrum occupancy mask), and spatial extent of the channel (through the propagation of the signal). This signal should be pseudo-random, coded so that signal-processing gain can recover the signal when it is weaker than the noise and interference. A filtered PSK PN sequence of 100 bits duration at a 10 k chips per second rate, filtered to an 8 1/3 kHz bandwidth would advertise an 8 1/3 kHz channel in a 10 ms burst. The offeror listens for 10 ms and then repeats the advertise-signal twice more. The sequence starts as close as practicable to the tick of the offeror's local GPS-second clock. The three-flag series repeats on the next second. A legacy user (or spectrum manager) could hear these bursts and realize that the channel is available for rent, expressing an objection by keying a transmitter.

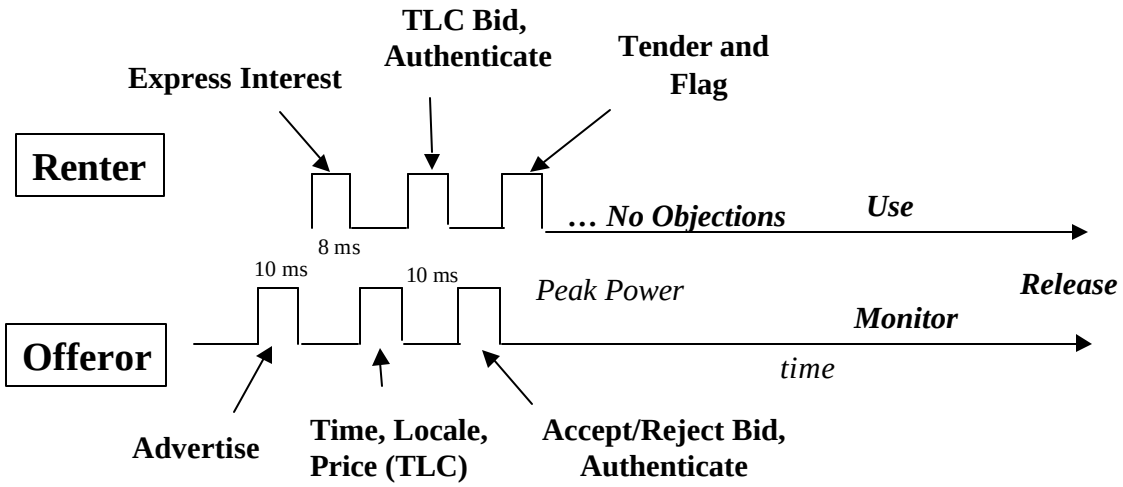


Figure III-1 Time Line of Spectrum Rental Protocol

A renter can express interest with a coded interest-burst similar to the advertise-burst. If the offeror hears the interest-burst, the second burst specifies the rental time interval, operating

locale, and price of the channel. This data exchange would be Huffman coded using a-priori knowledge. The renter then submits a bid with a short authentication sequence. The offeror may accept the bid, authenticating itself in return. Finally, the renter tenders the (e- or s-) cash, completing the initial rental protocol.

Both then wait and listen for the rest of this GPS second for objections. The 100-bit objection-sequence would be nearly orthogonal to the advertise- and interest-sequences. A legacy radio that begins to use the channel in native mode (e.g. FM push to talk) automatically negates the rental agreement if its received signal strength at the renter or offeror location exceeds a threshold. A 100-bit rental-cancellation sequence from either party then cancels the deal. The offeror cannot attempt to rent the channel again until after a specified waiting period (e.g. seconds to minutes), or until re-advertised by the offeror. After using the channel successfully, the renter provides the additional e-cash validation bits required to secure payment of the offeror's final bill. The initial rental protocol identifies the renter sufficient to pursue a claim if the renter defaults after using the channel. To avoid problems with the one-second granularity of the rental agreements, a service provider provisions the network by renting a few standby channels for traffic that cannot wait for the next rental period. During the use of the channel, both offeror and renter use a polite backoff protocol.

B. Polite Backoff Protocol: Defer to Authorized Legacy Users

If the cognitive radios using the channel employed a conventional air interface, legacy users would be unable to break in. Thus, for example, a police officer could not use his assigned frequency to call for assistance. The polite backoff protocol illustrated in Figure III-2 solves this problem, albeit at the expense of some loss of throughput.

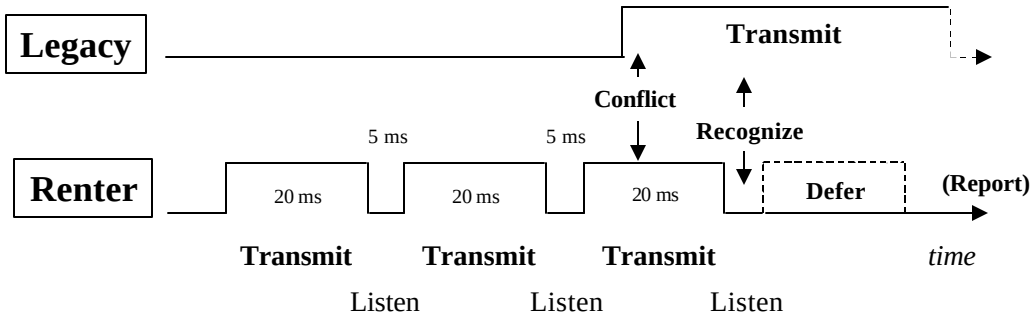


Figure III-2 Polite Backoff Protocol Assures Channel Access

The renter supplies (digital) traffic to the channel for 20 ms as shown in the time line. Both renter and offeror listen during the subsequent 5 ms listen-window. The high rate of listen-windows assures that not more than 25 ms of legacy speech would be truncated, a level that should be essentially imperceptible distortion of a push-to-talk radio signal. If a legacy waveform exceeds a carrier to interference ratio (CIR) threshold, for either renter or offeror, the conflict is recognized and the channel is immediately vacated. The truncating party issues a 10 ms termination-burst that indicates the cause is legacy interruption. The rest of the traffic would be sent on another channel. Clearly, the renter can claim that the goods were not delivered, and not send the final payment bits. The renter and the offeror would log the time, place and other

parameters of the legacy use of the channel so that spectrum managers could identify abusers. Since e-mail, attachments, file transfers, audio clips, video clips, and other asynchronous multimedia are relatively insensitive to end-to-end delay, this mode would be acceptable for low-cost wireless access. Of course, the offeror could make the channel available with peek-through required in only every Nth listen-window, introducing some clipping at the onset of reclamation of the channel. This enhances the throughput (and price, perhaps) accordingly. Police might be able to offer only the high listen-window mode. Fire departments or military users might be willing to key the microphone for a second or more to reclaim the channel. Once reclaimed, the channel remains dedicated to the legacy user either until the expiration of a pre-defined time-out or until the channel is again offered for rent. In addition, the full protocol would contain sequences that indicate the channel has already been rented (a "sold" sign).

C. Precedence and Priority

All users want guarantees that spectrum will be available when and where needed. Thus, any workable pooled spectrum approach has to have a way of providing such guarantees. Figure III-3 provides an example precedence of spectrum uses. The character of the existing band allocations defines the default spectrum-use precedence. Designated authorities may change precedence globally or locally. The etiquette allows one to designate a user (e.g. by international mobile subscriber identification), a channel, or any combination of [user x time x space x frequency] with a specific precedence.

1. **Emergencies** - *Established by authorities, inferred from events*
2. **Government** - *Attributed by band or channel modulation*
3. **Public Interest** - *Default by band, inferred from events*
4. **Commerce** - *Default by band and mode, inferred*
5. **Other** - *Recreational, sports, hobbies, etc.*

Figure III-3 Precedence of Spectrum Use

If these notions are subjectively acceptable, the task remains to formalize them so that the radio control algorithms will perform as intended. In particular, the radios have to be able to infer many aspects of precedence from events. This is a technical challenge that requires that the radios become context-aware. The formalism therefore must provide some strong evidence that the system as a whole supports the statutory guarantees in spite of the sharing arrangements. One mechanism that supports such guarantees is an order-wire for the coordination of needs and plans for spectrum use and assignment.

D. An Order-Wire System and Knowledge Exchange Language

The spectrum offeror may post an order-wire channel. An order wire is an ad-hoc signaling and control channel. If the offeror does not post one, the cognitive radios using the band could create one using a peer network in which the first user becomes the network control station (e.g. JTIDS[9]). The details of such a network are not central to cognitive radio research, but the language used to represent general world knowledge, plans, and needs is a key issue.

The Radio Knowledge Representation Language (RKRL) is the language and knowledge structure being used to develop cognitive radio at KTH. The Knowledge Query and Manipulation Language (KQML) [10] was explicitly designed to facilitate the exchange of such internal knowledge. Based on "performatives" such as "tell" and "ask", KQML readily express pooled-spectrum management information. KQML's ":content" tag, for example, delivers unstructured content. Although this general purpose tag would suffice, the introduction of new tags for spectrum pooling imparts additional structure to the dialog. The new tags include :Rental_offer, :RF_low, :Nchannels, :allowed_formats, :Legacy, :Equivalent, and tags for standard PCS formats such as DECT, GSM, and new 3G modes. The tags :From and :To refer to the time at which the rental is being offered. Using KQML, mobile nodes and networks may share plans about anticipated needs for spectrum so that it may be efficiently identified and rented. The KQML plan to offer spectrum uses the Tell performative to tell the (cognitive) network its plan as shown in Figure III-4. In this example, the radio also warns the network that its legacy users employ 25 kHz push-to-talk FM radios.

```
{Tell :language RKRL :ontology Spectrum Rental
      :Rental_offer (:Owner Fairfax_Police  :Location  Chantilly_VA
            (:RF_low 451 :Nchannels 12 :From 141118  :Until 141523)
            :Allowed_formats ((:DECT 32kbps) (:GSM GPRS) (:Equivalent))
            :Legacy 25kHzFM)}
```

Figure III-4 KQML Expression of a Plan

The ontology performative could invoke a special format for the ":content" part of a normal KQML message. The tagged format shown here is meant to suggest both a more widely endorsed set of standard tags, and the opportunity for significant data compression on such messages. The network uses the ontology to look up defaults and compressed codes for the general knowledge expressed in the packet. The other aspects of the plan are self evident, which is one of the strengths of KQML.

E. Related Topics

Cognitive radios need metrics for cost and value along with other high level rules of etiquette, such as:

1. "Tell the truth about who you are and what you need,"
2. "Block calls entering a disaster area and expedite those leaving," and
3. "Test evolving protocols in off-peak hours."

F. The Complexity of A Society of Cognitive Radios

Given a reasonable development of the spectrum pooling framework outlined above, one might think that performance could be projected using contemporary radio-engineering techniques. But cognitive radios will have complex internal structure and an ability to adapt to local circumstances. Since they also will be richly interconnected, they will form a complex adaptive system [11]. Cognitive radios could behave like an ant colony [12], evolving their own

paths through the spectrum and intervening nodes to ferry voice, data, video, and multimedia packets through the ether. These radio-ants might move packets from the smaller power-starved radios through multimode vehicular radios and on to conventional cell sites. If the etiquette is too strict, very little additional benefit will come from spectrum pooling because the control overhead will be too high to be workable. If, on the other hand, the etiquette is too liberal, there will be much interference and universally poor quality of service. Such complex adaptive systems operate best "at the edge of chaos" [11]. This is not a particularly comfortable place for spectrum managers. Such complex adaptive systems are in fact difficult to understand, model and diagnose [13]. Nevertheless, they also produce efficient answers to NP-hard problems [12]. Thus, in some sense an ant-colony of cognitive radios left to evolve spectrum use among themselves might be the most efficient way to achieve high value from limited radio spectrum in a reasonable time. How can we structure the capabilities and etiquettes of cognitive radio so that spectrum pooling is workable? There are many important aspects to this question. Cognitive radio research at KTH continues to address such questions.

IV. Conclusion

Software radios provide a vast untapped potential to personalize services. But the contemporary process of spectrum allocations takes years to decades and lacks flexibility. In part, this is because there is no reliable technology for guaranteeing spectrum use to its primary owners. This limits the flexibility and responsiveness of the radio to the network and to the user. Cognitive radio offers the opportunity to employ spectrum rental protocols in a way that is sensitive to users and to the communications context. The cognitive radio rental etiquette is thus offered as an approach to more efficient use of a limited resource that is in high demand. Its agent knowledge and inference mechanisms are under development, as is the initial critical-mass definition of RKRL. The use of KQML in the exchange of plans among cognitive radios appears promising. The goal of the cognitive radio research is to develop software agents that have such a high level of competence in radio domains that they may accurately be called "cognitive". The present research is merely a small step in this interesting research direction.

Acknowledgement

The author would like to thank Professor Gerald Q. Maguire and Professor Jens Zander of KTH, and Dr. Erland Wikborg of Ericsson for their insightful support of the research reported in this paper.

---

[1] Mitola, J., *Software Radio: Wireless Architecture for the 21st Century* (103255.2507@compuserve.com: Mitola's STATIS*faction* ISBN 0-9671233-0-5) 1999

[2] Mitola, J., "Software Radio Architecture: A Mathematical Perspective", IEEE Journal on Selected Areas in Communications (NY: IEEE Press) May 99

[3] SDR Forum Web Site (www.sdrforum.org) 1999

[4] Mitola, J., *Cognitive Radio – Model-based Competence for Software Radios*, Licentiate Thesis (Stockholm: KTH) September 1999.

[5] Negroponte, Nick (Cambridge, MA: MIT Multimedia Laboratory) circa 1990

[6] Tetra is a radio technology of Ericsson, Stockholm, Sweden (www.ericsson.se).

[7] Zander, J., "Radio Resource Management in Future Wireless Networks: Requirements and Limitations" IEEE Communications Magazine (NY: IEEE Press) August 1997

[8] Mikkonen, J., *Quality of Service in Radio Access Networks* (Tampere, Finland: Tampere University of Technology) May 1999.

[9] Ziemer and Petersen, *Digital Communications and Spread Spectrum Systems*, (NY: Macmillan) 1985.

[10] Finin, T., "KQML -- A Language and Protocol for Knowledge and Information Exchange," *International Conference on Building and Sharing of Very Large-Scale Knowledge Bases*, Tokyo, December, 1993.

[11] Kaufman, S., *At Home in the Universe* (NY: Wiley) 1995

[12] Dorgio and Gambardella "Ant Colony Systems: Cooperative Learning of the Travelling Salesman", IEEE Transactions on Evolutionary Computation (NY: IEEE Press) April 97

[13] Fogel, D., et al, "Inductive Reasoning and Bounded Rationality Reconsidered" IEEE Transactions on Evolutionary Computation (NY: IEEE Press) July 1999.

Appendix E: Cognitive Radio: Agent-based Control of Software Radios

Joseph Mitola III
Royal Institute of Technology (KTH), Stockholm, Sweden (Student) and
The MITRE Corporation, 1820 Dolley Madison Blvd, McLean, Virginia 22102 USA

***Abstract* - Abstract: Software radio technology expands the capability of radio nodes to multiband multimode RF configurations. Software-radio based Personal Digital Assistants (PDAs) will soon be capable of operating on second generation (2G) cellular, GPRS, EDGE, 3G, cordless telephone, RF LAN, Bluetooth, GPS, and/or other radio channels. The available bit rates, quality of service (QoS) parameters, and costs vary by orders of magnitude as a function of RF band, channel access mode, and propagation conditions. Such a bewildering array of alternatives renders user-selection of band and mode ineffective at best. Instead, cognitive radio technology will allow users to train PDAs to make context-sensitive choices for the user. This enhances the control technology of software radio by integrating a-priori knowledge, speech processing, text processing, pattern recognition, planning, and machine learning techniques in an extensible intelligent-agent framework. Ultimately, cognitive PDAs would learn the daily, weekly, monthly, and annual patterns of user communications needs. The detection of user communications-states is necessary for the autonomous selection of band, mode, format, and timing of communications tailored to user preferences and use-context. A Radio Knowledge Representation Language (RKRL) provides a-priori knowledge needed to bootstrap the learning algorithms. The XML version of RKRL offers a standard web-based ontology of radio context and architecture needed for effective coordination between the network and a wireless PDA. The cognitive radio prototype developed at KTH is described with initial test results.**

KEYWORDS: Software Radio, Digital Radio, Cognition, Model-based Reasoning, Machine Learning

I. INTRODUCTION

This paper concerns the application of auto-extensible intelligent agent technology to wireless service delivery. Cognitive radio is described, and a rapid-prototype cognitive radio, CR1, is discussed. The emergence of software-defined radio (SDR) as the platform of choice for 3G wireless motivates the functional objectives of cognitive radio.

A. Software Radio Evolution

Multimode radios [1] generate multiple air interface waveforms ("modes"). A mode may be partitioned into features of the modem, the protocol stack, and the network. Software-defined radio (SDR) modes implemented in a layered architecture support wireless services in a platform-independent way [2] as illustrated in Figure 1.

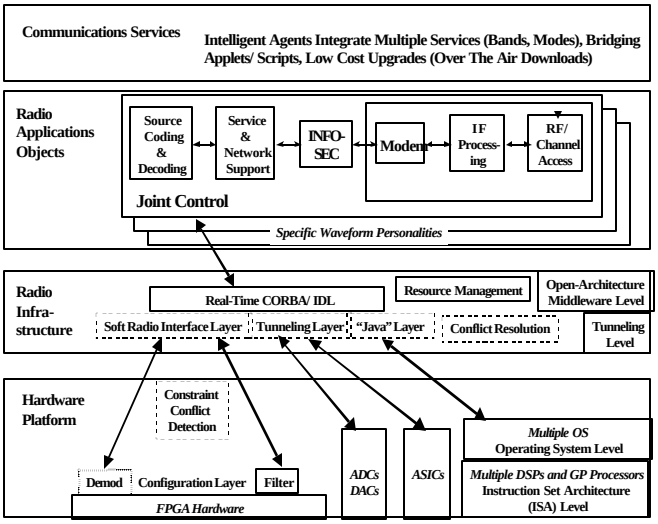


Figure 1 Layered Software Radio Architecture

Historically, wireless modem, protocol stack, and network aspects have been loosely coupled. High performance modems have been hardware-intensive with decades-long evolution of standards for channel coding. GSM, for example, was initially defined in 1983 [3] but not initially deployed until 1993. The 3G evolution for Asian markets may take eight years, in part because of the time required to reach international consensus [4]. The implementation of air interfaces in software and/or firmware-defined modems removes technology barriers to rapid evolution of waveforms. For example, GSM SDR base stations have been described [5, 6]. Recent research has shown that GSM, the Japanese PDC, and the North American IS-54/136 modems are readily implemented in a single parameterized software package (versus requiring a separate software package, RAM and ROM

per waveform) [7]. The simulation of baseband modems for DECT, GSM, and W-CDMA demonstrates the rapid development of waveforms customized to emerging services, such as data rates in excess of 32 kbps [8]. A packet-driven firmware architecture for implementing such alternate personalities in reconfigurable FPGAs has also been described [9]. The proliferation of air interface modes is illustrated in Table 1. The dimensions of the associated control parameter space include RF band (e.g. Cellular), mode (e.g. GSM, WCDMA), and mode parameters. Mode parameters include data rate, forward error control (FEC) mode, code rate, and QoS parameters (e.g. ATM cell loss rate, delay spread, etc. [10]). Mode parameters depend on environment parameters that include signal to noise ratio (SNR) or carrier to interference ratio (CIR). Within ten years, most wireless PDAs will be capable of accessing all the modes shown in Table 1.

Table 1 Forty Illustrative E-Mail Modes

Bearer	**Mode**	**Data Rate**	**Other Metric(s)**	**Remarks**
GSM	GPRS CS-1	9.05 kbps	Code Rate ½, 40 BCS	Best protection
GSM	GPRS CS-2	13.4 kbps	Code Rate 2/3, 16 BCS	Punctured (CS-2-4)
GSM	GPRS CS-3	15.6 kbps	Code Rate ¾	
GSM	GPRS CS-4	21.4 kbps	Code Rate 1	Highest throughput
IS-136 HS	Class C	<=384 kbps	Delay <1.5s; BER<10^{-6}	BER after ARQ
IS-136 HS	PCS 6, 5, 2, 1	69.2-22.8 kbps	Isochronism	But expensive
WCDMA	Low Rates	8, 16, 32 kbps	Code rate	All users
WCDMA	Medium	64, 144, 384	Cell loss rate for ATM video	Pedestrian
WCDMA	High Rates	384, 2048	Continuity of rate	Indoors
AMPS	CDPD	< 5 kbps	FEC Protected, low SNR	40 bytes offered
VHF	V-Series	0.075-9.6 kbps	Hybrid ARQ	Notional
UHF	V-Series	1.2-28.8 kbps	Hybrid ARQ	Notional
LMR	Proprietary	20 kbps	Hybrid ARQ	Notional
IEEE 802.11	2 Mbps	~0.5-1.84Mbps	300-8% overhead	40-1500 byte payload
IEEE 802.11	11 Mbps	~0.860-7.6Mbps	1179-31% overhead	40-1500 byte payload
HiperLAN/1	EY-NPMA	~1.52-15 Mbps	highest priority	40-1500 byte payload
HiperLAN/2	W-ATM	~7.4-16.67 Mbps	DSCx delay, FEC, ARQ/FEC	40-1500 bytes offered
DECT	Notional	32-155 kbps	Interleaved, FEC	Peak 892.4 kbps

Since processing capacity and memory density continue to follow Moore's law, within the same ten years, one may expect a battery operated commercial PDA to have a GFLOP of processing capacity. Memory should exceed 100 MB of RAM, with 2 GB of disk storage [11]. One theme of cognitive radio is to use this increased processing capacity and memory to reduce user interface complexity while facilitating the rapid insertion of new SDR technology (e.g. enhanced modem algorithms).

B. Environment-Aware Computing

The European Community's Advanced Communications Technology and Services (ACTS) program included a major thrust in location-aware computing, OnTheMove. In middleware called MASE, for example, a location manager determines location parameters and accuracy through the Global Positioning Satellite System (GPS) or through network facilities[12]. The mobile graphical user interface employs voice recognition for map navigation. Field trials of a City Guide [13] yielded high consumer interest in mobile email, a personal news service, personal stock portfolio, and maps. Other researchers investigated mobile video streams, clips, and fast file downloads [14].

Beadle, Maguire, and Smith [15] took these notions a step further. They developed environment-aware computing-communication systems. These include smart badges that use wireless technology to become aware of the immediate environment including doors, heating, ventilation, air conditioning, lighting, appliances, other computers, telephones, and pagers. Thermal, audio, and video sensors in a cognitive PDA would yield new opportunities for human-centric information services. Environment awareness takes

mobile computing, PDAs and networks beyond location-awareness.

Cognitive radio extends environment-aware computing to integrate machine learning to continuously tailor services to a user's communications context. In cognitive radio, communications context consists of time, place, and user state. Expression of context thus requires a standard radio ontology such as RKRL. It should mediate radio-related information exchanges among diverse radio-aware applications. Applications developers will not be expert in the myriad of bands and modes available to emerging SDRs. Yet each offers a different mix of QoS parameters. Thus, there is a need for a control agent that can use a radio ontology to manage access to radio bands and modes as a function of the application and user context. Cognitive radio is envisioned as a framework for developing such control agents.

C. Intelligent Agent Technology

Cognitive radio is an example of agent technology in telecommunications. The compendium, Intelligent Agents for Telecommunications Applications [16], and the feature topic on Mobile Software Agents for Telecommunications of the IEEE Communications Magazine [17] summarize the state of the art in applying automated reasoning to telecommunications. Mobility aspects are couched in terms of network applications of "autonomous, interactive, reactive" software objects. Goal-orientation, mobility, planning, reflection, and cooperation are described as additional attributes of agents that distinguish them from other types of software. Although wireless is mentioned in some of the papers, none of the contributions describes anything approaching cognitive radio. Moreover, none of the prior work indicates plans for a radio-specific taxonomy or radio domain language like RKRL.

Albayrak [18] defines an agent as one that exhibits the functional attributes of autonomy, interactivity, reactivity, goal-orientation, mobility, adaptivity; and that is capable of planning, reflection and cooperation. No requirement for machine learning is expressed. One thesis of cognitive radio is that a range of machine learning technologies is needed to autonomously tailor services to specific users [19].

Agent models include Aglet, Agent Tcl, Agents for Remote Access (ARA), Concordia, Mole, Odyssey, TACOMA, Voyager, and SHIP-MAI [20]. The focus has been on Internet applications, with minimal attention paid to the physical, data link and radio-related protocols that are critical to wireless. The Wireless Applications Protocol (WAP) facilitates the integration of mobile devices into such an agent framework [21]. Agent features of security, portability, mobility, agent communications, resource management, resource discovery, self-identification, control, and data management are also generally relevant to cognitive radio.

An open, web-based cognitive radio architecture also needs an agent coordination language like the Knowledge Query and Manipulation Language (KQML) [22]. High level architectures for sharing domain knowledge using KQML have included petro-chemical plant applications [23]. Reilly's comparison of intelligent agent languages is revealing [24]. He compares KQML, Java, TeleScript, Limbo, Active X and SafeTCL against his criteria for Intelligent Agent Managed Objects. These criteria include target environment, platform independence, execution style (e.g. interpreted versus compiled), native support for agent communications, agent mobility, and security features. Knowledge representation issues were not addressed explicitly. Most agents employ implicit models of user services and network capabilities. These models tend to be computationally informal, augmented with intuitively accurate definitions, but lacking a framework in computational linguistics or knowledge representation.

II. RKRL

Cognitive radio, on the other hand, builds on KQML as a knowledge representation language. Performatives from KQML are built into RKRL. In addition, RKRL includes model-based semantics for the radio and network domains. RKRL includes 4000 frames of XML [25] organized into about 40 micro-worlds.

A. Micro-Worlds

Formal methods employ an axiomatic treatment of a domain in order to prove theorems about that domain. In logic programming, for example, knowledge of the domain is represented in Horn clauses [26]. A theorem-prover resolves hypotheses. In the process, the system achieves goals, performs data base queries, and accomplishes other useful tasks. Alternatively, procedural knowledge may be structured as if-then rules [27,28]. The structure of domain knowledge in rule-based expert systems and logic programming is somewhat ad-hoc, driven more by the formalism (e.g. Horn clauses) than by the domain. In addition, the knowledge representation is tailored to the narrow task of the application, which is appropriate for the early development of such technology. One chronic problem of expert systems is the knowledge engineering bottleneck, the need for a knowledge engineer to

provide new rules for every new situation confronting the expert system. Cognitive radio is formulated as a machine-learning system in part to ameliorate such bottlenecks. Now that there are hundreds of logic programming and rule-based applications relevant to wireless information services, the focus of knowledge representation may shift from the application-specific aspects to cross-application aspects. One useful step is to organize the radio domain into an ontological system.

Research in naïve physics provides some useful insights. Davis describes a micro-world as a concept for organizing knowledge in the common-sense domain of cutting wood [29]. This entails the formalization of the eight components illustrated in Figure 2.

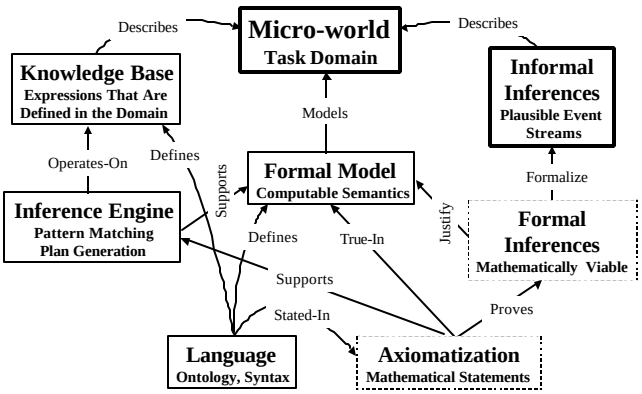


Figure 2 Davis' Micro-worlds Framework

The micro-world itself is the domain about which one is reasoning. In wood-cutting, objects occupy time and space; and cut each other, creating new objects. One of his formal models defines cutting in terms of "chunks." In addition, a knowledge base represents a-priori facts and procedures. The domain-specific language expresses the formal model. In addition, a formal inference system defines the formal subset of an identified set of informal inferences that may be drawn about the micro-world. The language relies on an axiomatization, which defines "truth" in the formal model. This axiomatization supports proofs induced by the formal inference system.

Radio knowledge, similarly, may be organized into micro-worlds [19]. Radio is a large, complex domain in which competence with bands, modes, and protocols requires a large set of axioms. If structured into a single micro-world, axioms about GSM "channels" would conflict with axioms about IS-95 "channels." RKRL therefore structures wireless into a set of interacting micro-worlds within each of which an appropriate taxonomy is defined.

B. Axiomatization of RKRL Micro-Worlds

Communications contexts lack mathematical structure. However, one may induce topological spaces on radio architecture (functions, components, and design rules) [30]. The analysis of these topological spaces underlies the definition of RKRL for cognitive radio. It specifies the sets and subset relationships among the radio entities that establish generalized communications context. It also defines conceptual and computational models applicable to subsets of the domain. RKRL is an incrementally structured taxonomy for wireless services. It does not impose the structure of a strict hierarchical taxonomy, but it describes the set-theoretic structure to the degree that it has been articulated for the domain. An RKRL frame is an expression:

<frame> :=

<handle><model><body><context><resources>

For example, the fact that a user's address includes a city is expressed in RKRL as: ***<handle> Address </handle> <model> contains </model> <body> City </body>.*** Note that the XML dialect structure uses <tag> end tag </tag> to denote RKRL tags.

"Contains" is the primary set-theoretic model that structures the radio domain. In addition, the context element provides explicit paths from a universal root to all conceivable entities, e.g.:

<context>Home/ User/ Person/ Concepts/ Universe/ RKRL1.0 </context>

The /Concepts/ subset defines classes of entities, while instances have their own space-time context such as:

<context>Joe'sHome/ Fairfax/ VA/ USA/ Global Plane/ Physical world/ Universe/ RKRL1.0</context>

Extended context includes the place, time, and source of radio-related information. Knowledge in RKRL is organized into the forty micro-worlds listed in Table 2.

The spatial ontology sets the earth in the universe with earth-orbiting communications satellites. The global plane defines continents, oceans, and countries of the world needed for reasoning about air travel. This strategy iterates down to the local level at which the GPS location of buildings is known from a street map.

In the immediate plane, radio signals from the refrigerator operating on a smart-kitchen RF LAN define the spatial relationships and information resources of each local entity.

Table 2 RKRL Micro-Worlds

Constellation	Micro-worlds
Meta Level	Globals, Inference Engines, Meta, Cognition, Goals, KQML, KIF, Skills, Sets, Database, Parsers

Ontological	Universe, Self, Concepts, Models, Time, Space, Spectrum, User
Spatial	Global, Satcom, Regional, Metro, Local, Immediate
Radio Structure	Architecture, Functions, Internal, Hardware
Software	Software, CORBA, UML, ODP, MPI
Wireless Functions	Air, Modem, Demod, Equalizer, Memory
Protocols	Protocol, Physical, Data Link
Networks	Network, Cellular, Segmentation, Messages, Propagation
Other	References

At KTH, this approach has been used for wireless broadcast of location and status (e.g. of doors: open, closed) to smart badges [15]. One aspect of the current research is to use RKRL to structure knowledge in a cognitive radio prototype, CR1.

III. Cognitive Radio

Cognitive radio is a framework for evolving auto-extensible intelligent agents to control radios.

A. *Framework*

Software radios have considerable computational capacity, but little cognitive ability. For example, a GSM SDR's equalizer taps reflect the channel impulse response. If the network wants to ask today's handsets "How many distinguishable multipath components are in your location?" two problems arise. First, the network has no standard language with which to pose such a question. Second, the handset has the answer in the structure of its time-domain equalizer taps internally, but it cannot access this information. It has no computationally accessible description of its own structure. Thus, it does not "know that it knows." It cannot tell an equalizer from a vocoder. To be termed "cognitive," a radio must know basic facts about radio. It should know that an equalizer's time domain taps reflect the channel impulse response. A cognitive radio contains a computational model of itself including the equalizer's structure and function. It uses RKRL to accomplish this as suggested in Figure 3.

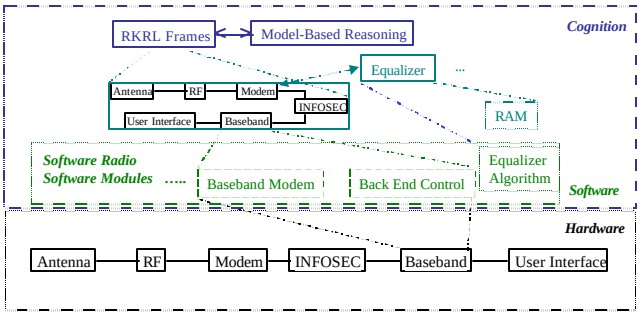


Figure 3 Cognitive Radio Framework

The radio hardware consists of modules: antenna, RF section, modem, INFOSEC, baseband processor and user interface. In the example, the baseband processor hosts the software. This software includes the baseband modem and equalizer module. In addition, however, a cognitive radio also contains internal models of its own hardware and software structure. The model of the equalizer, for example, could contain the codified knowledge about how the taps represent the channel impulse response. The RKRL frames plus a related extensible model-based reasoning capability, give the radio its "cognitive" ability. The radio's model of itself contains models of its components (antenna, RF conversion, etc.). In addition, the interfaces among these components are modeled. The SDR Forum has defined these interfaces using the CORBA IDL [31]. This provides a starting point. The Forum has not defined a formal semantics that could mediate machine reasoning about these interfaces, however. RKRL 1.1 begins to fill this void. It extends the SDR Forum approach with formal semantics. This includes logical sorts and axioms for time, space, and user identity. It also includes the set-theoretic formulation of radio in the RKRL micro-worlds. Cognitive radio employs RKRL via the cognition cycle.

B. *The Cognition Cycle*

A cognitive radio agent interacts with its environment via the cognition cycle of Figure 4. Stimuli enter the cognitive radio as interrupts, dispatched to the cognition cycle for a response. The cognitive radio continually observes its environment, orients itself, creates a plan (s), decides, and then acts. In addition, a cognitive radio employs machine learning techniques throughout. This includes learning its own personality by being trained both a-priori and by its user. This also includes learning human and machine protocols as it is exposed to them experientially.

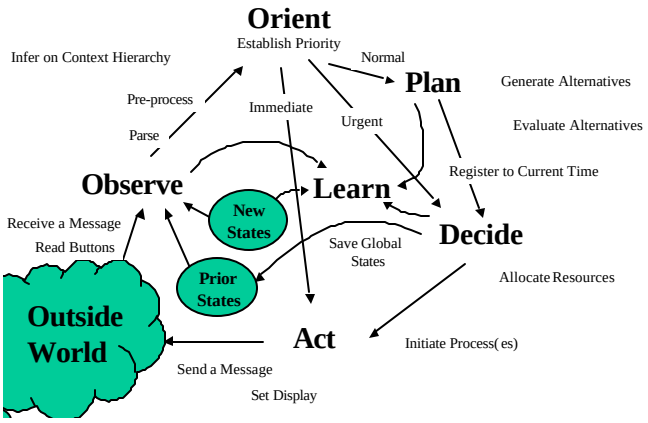


Figure 4 Simplified Cognition Cycle

The receipt of a new message constitutes a new stimulus, initiating a new cognition cycle. The cognitive radio parses all stimuli to identify the communications context. In the CR1 rapid prototype, stimuli are organized into reinforced hierarchical sequences in this observation phase. It then orients itself by determining the priority associated with this stimulus. A power failure might directly invoke an act ("Immediate" path in the figure). A non-recoverable loss of signal on a network might invoke reallocation of resources ("Urgent" in the figure.) However, an incoming network message would normally be dealt with by generating a plan ("Normal" path). Planning consists of plan generation. The "Decide" phase selects among candidate plans. The radio might have the choice to alert the user to the incoming message (e.g. behaving like a pager) or to defer the interruption until the current business meeting is over (e.g. behaving like a secretary who is screening calls). "Acting" initiates processes performed by other radio software, such as dialing a number on the GSM network. Learning is a function of observations and decisions. For example, prior and current internal states may be compared with expectations to learn about the effectiveness of a communications mode.

This cycle implies a large potential scope of hard research problems. Parsing incoming messages requires natural language text processing technology. Scanning the user's voice channels for content that defines the communications context requires speech processing. Orienting to user context may require conceptual clustering [32], or an extension of document clustering [33] that works well on small, imbalanced data sets. Planning technology offers alternatives in temporal calculus, constraint-based scheduling, task planning and the like [34, 35, 36]. Resource allocation itself includes algebraic methods for wait-free scheduling protocols, Open Distributed Processing (ODP), and Parallel Virtual Machines (PVM), just to name those that have obvious relevance. Finally, machine learning remains one of the core challenges in artificial intelligence research. The focus of cognitive radio research, then, is not on the development of any one of these technologies per se. Rather it is on the higher-level data structures and processes that integrate the contributions of these diverse disciplines. The range of potential capabilities of such a cognitive radio is summarized in Figure 5.

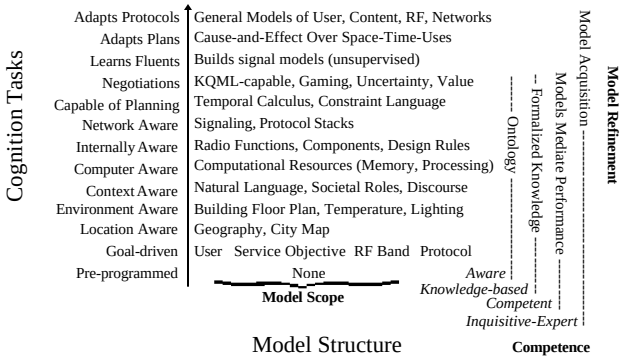


Figure 5 Cognitive Radio Representation Space

C. *Control Parameter Space*

Goal-driven choice of RF band, air interface, and protocol requires reasoning in at least two dimensions. Source parameters define the information structure of the source bitstream. Mode selection parameters define the tolerance of the source bitstreams to errors and other impairments (Table 3). Source parameters have been studied extensively [37]. In addition, Quality of Service (QoS) metrics for the wireline system have been normalized for wireless networks [38]. A simple goal-driven radio could be outfitted with tables of service-quality parameters as a function of band and mode for the bands and modes it can access. When an additional air interface is downloaded to the radio, its service parameters could be downloaded as well. Again, in principle, the goal-driven radio could select the band and mode that is available and that best meets the source requirements.

Wireless decision parameters vary temporally with location, time of day, exact frequency subband, traffic elsewhere in the network, weather, and other contextual parameters. There are daily and weekly patterns of business, sports and leisure that shape demand and hence congestion of bands. They also vary cyclically with month of the year.

Table 3 Mode Selection Decision Parameters

Data Structure	**Parameter**

Source Bitstream	Bit Rate
Burstiness	Constant (CBR) or variable (VBR)
Isochronism	None, real-time or near-real-time
Bursts	Maximum burst size
Tolerance	Parameter mismatch
Service Quality	Error rates
Bit error	BER, symbol error rate
Delay-related	Transfer delay, variance, jitter
Buffer-related	Packet or Cell Loss Rate (CLR)
Reliability	Grade of Service [P_{link}]
Assuredness	Authentication, privacy
Cost	Peak, off-peak, service-related

Table 4 Context Parameters

Data Structure	**Parameter**
Traffic Structure	Statistics of offered load
Temporal Cycles	Diurnal, weekly, annual, other
Location	Propagation, services
Bands/ Modes	RF range, mode parameters
Weather	Precipitation, road conditions
Immediate relative position	In pocket, building, on the ground
Orientation	Direction of travel
Travel Mode	Auto, train, air, foot: speed
User Profile	Identity, use patterns
User speech profile	Topics of discourse
User data profile	Topics of interest
Immediate needs	Context qualifiers
Connectivity needs	Criticality of connectivity
Security profile	Authentication, encryption
Intended recipient(s)	Generic, class, instance

Cognitive radio adapts its use of spectrum bands and modes by acquiring its user's space-time patterns. Its parameters for context-driven planning are summarized in Table 4. Mode of travel, for example, may be inferred through monitoring fade rates, which are different for pedestrian versus vehicular travel. Movement implies constraints on RF band, mode, and QoS. The transformation of radio mode and communications context parameters into actions is accomplished using an internal knowledge representation hierarchy of reinforced sequences.

IV. Reinforced Hierarchical Sequences

CR1 represents knowledge as reinforced hierarchical sequences as illustrated in Figure 6. These consist of the sensor data partitioned into generalizations of words, phrases, dialogs, and scenes. The partitioning is based on both time elapsed between elements of the sequence and on content-derived cues such as a restart signal in a framing sequence or the phrase "Lets start over" in a speech stream. A context is the sequence that results when the content of multiple sensor streams is co-joined at some level of the hierarchy. Each sequence at one level becomes an element at the next level. The time delay between successive occurrences is noted for use in sequence binding, a form of cross-correlation.

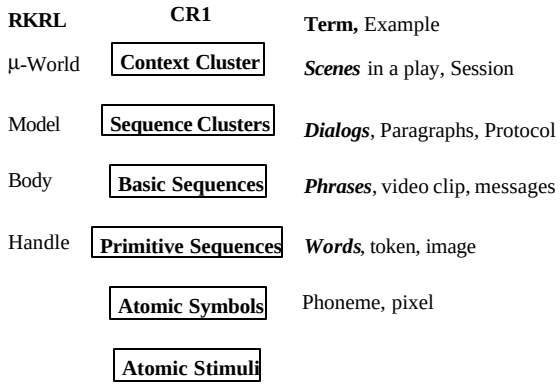


Figure 6 The Concept Hierarchy of Reinforced Sequences

Counting occurrences reinforces experience. In addition, user actions such as training, positive, and negative reinforcement change counts as if the experience had occurred more often or not at all. Sequence elements vary from the atomic to the complex as a function of the level of the concept hierarchy. Atomic stimuli (e.g. from a microphone) are converted to atomic symbols (e.g. phonemes) by preprocessors such as speech-to-text phoneme recognizers. Temperature, pressure, time, location, and radio-channel states determined by lower level software-radio personalities are formed into phrase and higher level contexts. Reasoning is the process of mapping sequences of external stimuli to internal sequences that represent world models, plans, and actions. Learning is the process of acquiring new sequences and associating internal and/or external actions with these sequences. The internal sequences follow the RKRL schema of <handle><model><body>, where <handle> is the stimulus that results in the response <body> when interpreted using <model>. This is denoted:

srModel: <stimulus> > <response>

Multiple srModels are organized into processing structures

following RKRL. Figure 7, for example, shows the models that support reasoning about modem parameters.

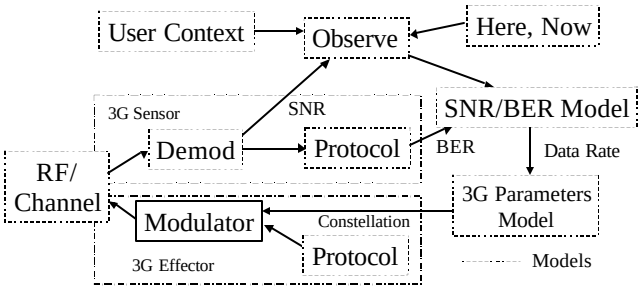


Figure 7 Model Structure to Set Modem Parameters

Sequences in models are bound to each other such that elements of established strings act as variables to which elements of new stimulus strings are bound. In terms of theory of languages and computing, these linked models comprise a finite state language (similar to Post productions), but with millions of states, which is equivalent to a push-down automaton (context free language) for (small) finite stack depth. This reasoning structure turns out to be equivalent to forward chaining, but not to resolution theorem proving, which is as intended. This specifically non-Turing capable approach meets the theoretical constraints of bounded recursion necessitated by the isochronous nature of communications [30]. It also substantially simplifies machine learning. Yet it is powerful enough to detect user communications states in scenarios like introductions and eating out [39].

A. Learning Artificial Protocols

The use of reinforced hierarchical sequences may be illustrated through an example of how CR1 learns and uses an artificial protocol. CR1's GSM sensor delivers text strings to a phrase-level parser. Simulated time, location, RF LAN, weather, (perfect) speech and text stimulus channels provide additional context. Suppose the following KQML-like string is presented on the text channel:

Plan :Location Stadium :CIR <9dB :GPRS-data-Rate 8.55 kbps

This could be the body of a KQML tell-one performative from the network telling the PDA to reduce its data rate in mode GPRS CS1 (from the nominal 9.05 kbps) in the cell site near the Stadium, for example. CR1 parses the string into:

p.l.a.n| :.L.o.c.a.t.i.o.n, S.t.a.d.i.u.m | :.C.I.R, <.9.d.B | :.G.P.R.S.-.d.a.t.a.-.R.a.t.e, 8…:value 5.5, k.b.p.s

In this sequence, dots (.) delimit characters within words, commas (,) delimit words within phrases, and vertical bars (|) delimit phrases within dialogs. CR1 knows in advance that "plan" is the name of its internal model **p.l.a.n** via its word-role srModel where **p.l.a.n > m.o.d.e.l**. CR1 implements such models as hash tables for O(1) access to the millions of sequences it must store for reasonable performance. The Orient phase composes a command string for this sequence:

m.o.d.e.l @ now @here | p.l.a.n | :.L.o.c.a. … (etc)

This model-update sequence explicitly includes time, location, and other relevant context. The plan phase recognizes this as a request to update an internal model here and now. The Decide and Act phases implement the change. The srModel for planning now includes:

(L.o.c.a.t.i.o.n, S.t.a.d.i.u.m, :.C.I.R, <.9.d.B, :.G.P.R.S.-.d.a.t.a.-.R.a.t.e) > 8…5.5, k.b.p.s

In addition, Orient phase model for new words now knows the words location, stadium, CIR, etc. After a sleep cycle, these words become integrated into the store of actionable information.

B. Applying Protocol Knowledge

Consider the plan required to set modem parameters:

Plan: :setParameters GPRS :value "Data Rate @CIR"

As before, the sequence has been installed in the **p.l.a.n** model. The planning goal is to complete the sequence of the plan, forwarding the resulting plan to the Decision phase for action. To use this plan, the stimulus :Send email arrives from the user (e.g. on the speech channel), and the GPRS mode is selected (using a different plan sequence stored in the same planning srModel. The planner then sets GPRS parameters using the plan shown above. It applies each phrase of the plan to its **p.l.a.n** model, aggregating the results into a final plan sequence. The sub-sequence @CIR yields <9dB because the Demod node of Figure 7 reports CIR by installing the srModel "@CIR > <9dB" into **p.l.a.n.** The completed sequence "Here, Now, GPRS Data Rate <9dB" yields (@GPRS Data Rate , 8.55 kbps) near the Stadium and 9.05 dB elsewhere. The tag @ in this sequence causes the body to be placed in **p.l.a.n,** essentially binding a local variable. **@,G.P.R.S D.a.t.a R.a.t.e > <.9.d.B** is available to subsequent planning steps. This sequence binding process continues yielding the final plan "GPRS setParameters Data Rate 8.55 kbps "

C. Sequence Correlation

A more natural way of expressing the plan given above is as follows:

"The plan to set-the parameters for GPRS is as follows. First, determine the CIR. Next, determine the mode for this CIR. Finally, set GPRS to this mode."

This plan reads like (tutorial-style) natural language. The phrase "is as follows" provides linguistic structure but no essential content. Sequence templates are artificial sequences that CR1 may be taught to set the structure of knowledge to be acquired. One of the sequence templates for a plan is:

"The plan to set x for y is as follows. First, bind the y. Next, bind the z for this y. Finally, set x to this z.

CR1 correlates each new sequence against all previously observed sequences in parallel, using the set-theoretic framework of RKRL. Each element, α, in a sequence, γ, is stored as a member of the set consisting of all sequences $\{\gamma\}_\alpha$, containing α. Correlation first intersects $\{\gamma\}\alpha_i$ for $\alpha_i \varepsilon \gamma_n$, the new sequence. The resulting $\{\gamma\}$ are then registered to γ_n according to a dynamic time warping algorithm [40]. An α_i in γ_j registers to α_j in γ_n if i =j. In addition, the reinforcement of γ_j is accrued as value. If $i \neq j$, then warping cost of the number of element delays times a weight is accrued. The best correlation is that which has maximum accrued (value-cost). Whenever α_i and α_{i+2} register, α_{i+1} is bound to one of:

(1) null, if no α in γ_j corresponds to α_{i+1},

(2) $\Sigma(\alpha_{i+1}{}^j \ldots \alpha_{i+m}{}^j)$, forming a sub-sequence

In addition, α_{-1} is a possibly empty prefix and α_{N+1} is a possibly empty suffix, bound in the obvious way. Sequence correlation on the above yields bindings:

x = "the parameters";
y = GPRS; z = mode;
bind = determine.

The bound elements act like function-variables, taking on the roles of the symbols to which they are bound (possibly with user confirmation). This is accomplished by recording the srModel for, "determine" that was previously stored for "bind." This simple reasoning by analogy allows the system to build a variety of internal models given a minimal initial set. With a good user interface, the system would ask the user for permission to treat "determine" as "bind" in the given context. Although it does not appear difficult to construct a generalization algorithm that operates on collections of such contexts, CR1 implements only the sequence correlation and binding operations for machine learning, lacking the ability to generalize across contexts.

Although the plan template and the pseudo-natural language plan are far from unconstrained natural language [41], they are both readily readable by people and also parsed and executed precisely by CR1. Moreover, they are assimilated in a way that identifies whether the system knows how to act on each word and phrase. Filling in missing models is the paradigm for interactive knowledge acquisition. Thus, although the natural language processing approach is relatively primitive, it serves the useful purpose of streamlining the automatic acquisition of machine and human contexts and protocols ..

V. CONCLUSIONS

Cognitive radio provides an interesting framework within which to experiment with the contributions of natural language processing and machine learning on the evolution of extensible intelligent agents for software radio. Such inter-disciplinary research often raises many more questions than it answers, particularly among experts in each sub-field. There are many important approaches to natural language processing and machine learning that might have been integrated into CR1. This present contribution suggests the potential benefits of these technologies to software radio. Those that have been integrated into CR1 seem to be just the "tip of the iceberg" of an interesting research area.

VI. REFERENCES

[1] Upmal and Lackey, "SPEAKeasy, the Military Software Radio" IEEE Communications Magazine (NY: IEEE Press) '95

[2] J. Mitola, "Software Radio Architecture Evolution: Foundations, Technology Tradeoffs, and Architecture Implications (Invited Paper)" IEICE Transactions on Communications, Special Issue on Software Radio Technology (to be published in 2000).

[3] Mouly & Pautet, "Evolution of the GSM System" IEEE PCS Magazine (NY: IEEE, Oct 1995).

[4] S. Blust, "IMT-2000: Third Generation Wireless, The Vision of a Global Wireless Network" (Tempe, Az: SDR Forum) 17 March 98

[5] T. Turletti, et al., "Towards the Software Realization of a GSM Base Station" IEEE JSAC Feb 99

[6] Zangi, K., "Software Radio Issues in Cellular Base Stations" IEEE JSAC (NY: IEEE Press) April 1999

[7] A. Wiesler and F. Jondral, "Software radio structure for second generation mobile communications systems" Proceedings of the IEEE Vehicular Technology Conference (NY: IEEE Press) 1998

[8] A. Wiesler, R. Machauer, and F. Jondral, "Comparison of

GMSK and linear approximated GMSK for software radio" Proc. of ISSTA5, (NY: IEEE Press) 1998
[9] Srikanteswara, S., et al, "A Soft Radio Architecture for Reconfigurable Platforms" IEEE Communications Magazine (NY: IEEE Press) Feb 2000
[10] Mikkonen, J., *Quality of Service in Radio Access Networks* (Tampere, Finland: Tampere University of Technology) May 1999.
[11] 1997 SIA Semiconductor Technology Roadmap and updates (www.semichips.org: Semiconductor Industries Association) Feb 2000
[12] Meggers, J., et al, "Mobile Multimedia for Small Handheld Devices", Proc. ACTS Mobile Summit, (Rhodes, Greece: The European Commission) June 98
[13] Kreller, B., "A Mobile-Aware City Guide Application," Proceedings of the ACTS Mobile Summit, (Rhodes, Greece: The European Commission) June 98
[14] Xiaohan Yu, "Mobile Multimedia Services Based on GSM HSCSD, " Proc. ACTS Mobile Summit, (Rhodes, Greece: The European Commission) June 98
[15] Beadle, H. W., Maguire, G. Q., and Smith, M. T., *Environment Aware Computing and Communications Systems*, (www.kth.edu: Royal Technical Institute) 1998
[16]Albayrak, *Intelligent Agents for Telecommunications Applications* (Amsterdam: IOS Press) 1998
[17] A. Karmouch,, "Guest Editorial: Mobile Software Agents for Telecommunications" *IEEE Communications Magazine* (NY: IEEE Press) July 1998
[18] S. Albayrak, "Introduction to Agent Oriented Technology for Telecommunications" *Intelligent Agents for Telecommunications Applications*, Edited by S. Albayrak (Amsterdam: IOS Press) 98
[19] J. Mitola, *Cognitive Radio: Model-Based Competence for Software Radio* Licentiate Thesis (Stockholm: KTH) 1999
[20] Anh Pham, V., and Karmouch, A., "Mobile Software Agents: An Overview," IEEE Communications Magazine (NY: IEEE Press) July 1998
[21] www.wapforum.org
[22] Finin, T., "KQML -- A Language and Protocol for Knowledge and Information Exchange," *International Conference on Building and Sharing of Very Large-Scale Knowledge Bases*, Tokyo, December, 1993.
[23] Chao, K.-M., et al, "Sharing of Domain Knowledge" *Intelligent Agents for Telecommunications Applications*, Edited by S. Albayrak (Amsterdam: IOS Press) 98
[24] Reilly, J., "Secure Intelligent Agent Extensions to the TMN Management Framework" *Intelligent Agents for Telecommunications Applications* edited by S. Albayrak (Amsterdam: IOS Press) 1998
[25] Extensible Markup Language (XML) (www.w3c.org) Dec 1999
[26] J. Minker, Logic Programming Course Notes (College Park, MD: University of Maryland) April 85
[27] N. Nilsson *Principles of Artificial Intelligence Programming* (Menlo Park, CA: Tioga Press) 1978
[28] Winston, P., Artificial Intelligence (Cambridge MA: MIT Press) 1982
[29] Davis, Ernest, "The Naïve Physics Perplex", AAAI Magazine (Palo Alto, CA: The American Association for Artificial Intelligence), Vol 19, No. 4, Winter 98
[30] J. Mitola, "Software Radio Architecture: A Mathematical Perspective" IEEE Journal on Selected Areas in Communications (NY: IEEE Press) April 99
[31] SDR FORUM TECHNICAL REPORT 2.0 Architecture and Elements of Software Defined Radio Systems as Related to Standards (Rome New York : SDR Forum) June 1999.
[32] R. Michalski "Learning from Observation: Conceptual Clustering" in *Machine Learning* (Palo Alto, CA: Tioga Publishing Company) 1983
[33] M. Sahmi et al, "SONIA: A Service for Organizing Networked Information Autonomously" Digital Libraries 98: Proceedings of the Third ACM Conference on Digital Libraries. (Association for Computing Machines) 98
[34] R. Wilensky, *Planning and Understanding* (Berkley, CA: Addison-Wesley) 1983
[35] S. Das et al, "Decision making and plan management by autonomous agents: theory, implementation and applications" Autonomous Agents 97 (ACM) 1997
[36] S. Hamid and N. Vaidya, "Log Time Algorithms for Scheduling Single and Multiple Channel Data Broadcast" MOBICOM 97 (ACM) 1997
[37] Stallings, Handbook of Computer-Communications Standards, Volume1, The Open Systems Interconnection (OSI) Model & Related Standards (NY:Macmillan, 1987).
[38] Mikkonen, J., Quality of Service in Radio Access Networks, PhD Dissertation, (Tampere, Finland: Tampere University of Technology) May 99
[39] J. Mitola III, Cognitive Radio, Doctoral Dissertation (Stockholm: KTH) March 2000
[40] Yfantis et al "On time alignment and metric algorithms for speech recognition" Proceedings, Conference on International Information Intelligence and Systems (NY: IEEE Press) 1999
[41] J. Allen *Natural Language Processing* (Redwood City, CA: Underwood Press) 1995