ACROPOLIS
European NoE on Cognitive Communications

SEVENTH FRAMEWORK PROGRAMME

Advanced coexistence technologies for radio optimisation in licensed and unlicensed spectrum

(ACROPOLIS)

Document Number D12.2

Policy Framework, Policy Enforcement Mechanisms and Hardware Constraints

Contractual date of delivery to the CEC:	31/03/2012
Actual date of delivery to the CEC:	31/03/2012
Project Number and Acronym:	257626 - ACROPOLIS
Editor:	Dionysia Triantafyllopoulou (UoS)
Authors:	Dionysia Triantafyllopoulou, Klaus Moessner (UoS), Daniel Denkovski, Valentin Rakovic, Vladimir Atanasovski, Liljana Gavrilovska (UKIM), Adrian Kliks (PUT), Nikos Dimitriou, Andreas Zalonis (IASA), Jing Lv, Eduard Jorswieck (TUD), Aimilia Bantouna, Vera Stavroulaki, Yiouli Kritikou, Kostas Tsagkaris, Panagiotis Demestichas (UPRC), Kamil Chudas (EIT+)
Participants:	IASA, TUD, UPRC, EIT+, UKIM, PUT, UoS
Workpackage:	WP12
Security:	Public (PU)
Nature:	Report
Version:	1.0
Total Number of Pages:	71

Abstract:

The purpose of this deliverable is to provide a summary of the scope, functions and basic mechanisms of the ACROPOLIS decision making framework. Its main focus is on policy based decision making and emphasis is given on the architecture, features and characteristics of the policy frameworks with respect to cognitive radio networking. It also describes the ACROPOLIS notion of a generic policy framework and provides guidelines for the use and implementation of policies in cognitive radio systems, highlighting the importance of decision making. Special attention is paid on the requirements and the constraints of the implementation of a policy framework on a hardware instantiation platform.

Keywords: Policy Framework, Policy Enforcement, Hardware Constraints

Document Revision History

Version	Date	Author	Summary of main changes
0.1	04.02.2010	UoS	Initial structure of the document
0.2	05.09.2011	UoS	Updated ToC
0.3	03.11.2011	UoS, UKIM, PUT, IASA	Initial Contributions
0.4	08.11.2011	UoS	Restructured ToC
0.4	25.11.2011	UKIM	Modified the existing contributions and added new ones in section 2.2.3 and 3.2
0.4	30.11.2011	EIT+	Updated section 2.1
0.4	30.11.2011	IASA, TUD	Updated contributions in section 4.1
0.4	30.11.2011	UoS	Updated contributions in section 2.2
0.4	06.12.2011	UPRC	Provided contributions in sections 3.1 and 3.3
0.4	22.12.2011	UPRC	Provided contributions in section 2.1.2
0.5	06.01.2012	UoS	Integrated input from all partners, Restructured ToC, Included Abstract, Introduction and Conclusions
0.5	16.01.2012	UKIM	Provided contributions in section 3.1.3
0.5	08.02.2012	UPRC	Updated contributions in section 2.1.2
0.6	08.02.2012	UoS	Updated Introduction, Integrated contributions
0.7	13.02.2012	UKIM	Restructured ToC, Updated Section 4
0.7	17.02.2012	PUT	Provided contributions in section 2.2.4, 2.2.5 and 3.2
0.7	20.02.2012	UoS	First complete draft version
0.8	13.03.2012	UoS	Integration of comments from IASA, UKIM, PUT and EIT+. Version submitted to the ExecCom for review.
0.9	27.03.2012	UKIM	Updated Section 4
0.9	27.03.2012	UoS	Provided Executive Summary, Addressed review comments
1.0	30.03.2012	UoS	Final Version

Executive Summary

Deliverable D12.2 is the second deliverable of ACROPOLIS Work Package (WP) 12. Its main purpose is to provide a summary of the scope, functions and basic mechanisms of the ACROPOLIS decision making framework that is responsible for the analysis, decision and spectrum allocation processes. Its main focus is on the architecture, features and characteristics of the policy frameworks with respect to Cognitive Radio (CR) networking. It also provides guidelines for the use and implementation of policies in CR systems, highlighting the importance of decision making. Special attention is paid on the requirements and the constraints of the implementation of a policy framework on a hardware instantiation platform.

The deliverable is organized as follows:

Section 2 first provides an overview of the existing and most notable policy frameworks for CR networks, namely the ARAGORN, End-to-End Reconfigurability (E2R), End-to-End Efficiency (E3), ORACLE, and IEEE 802.22 Wireless Regional Area Network (WRAN) frameworks. The main architectural components of each framework are summarized, giving an understanding of their merits and usefulness in CR networks. The section continues with the review of the most well-known policy languages for CR systems, namely neXt Generation Policy Language (XGPL), ORACLE Policy Language (PL) and Cognitive Radio Language (CoRaL), describing their basic structures and providing exemplary policy rules. Also, it briefly describes the requirements for policy languages that are provided by the recently introduced IEEE P1900.5 Standard. Finally, it makes a short discussion on the ACROPOLIS approach regarding the expression of policies to be used in the decision making framework to be developed.

Section 3 focuses on well-known policy framework standards and begins by providing a high-level overview of the IEEE P1900.4 Standard, describing its system architecture, the policy and decision making process, and the information flow between the different system entities. Then, it continues with a description of the IEEE P1900.5 Standard, providing the general architecture requirements for the policy-based control of Dynamic Spectrum Access (DSA) radio systems, followed by the description of its architectural components and interfaces.

Section 4 discusses the ACROPOLIS notion of a generic policy framework. Additionally it discusses on the deployment of policies in a CR system, paying special attention on the policy reasoning and decision making process.

Section 5 describes the general requirements and constraints of a CR instantiation platform and elaborates on the applicability of the Universal Software Radio Peripheral 2 (USRP2) platform on policy based CR scenarios, discussing on its features, hardware constraints, and the policy implementation issues. It aims at providing a link between the policy frameworks discussed in the previous sections and the operation characteristics of the actual experimental instantiation platforms, highlighting the requirements that have to be taken into account in order to implement a policy framework within a hardware platform.

Finally, Section 6 provides a summary and some concluding remarks of this deliverable.

Table of Contents

1. Introduction

Policies are sets of rules that are used in a network in order to control the behaviour of the nodes and manage the available resources. Thus, they can be adopted by a CR network, exploiting the CR – specific characteristics and capabilities of the different network entities, in order to enhance spectrum utilization and radio resource usage. Their purpose is to achieve specific goals related to the optimal and most efficient use of the available network resources. To guarantee optimization of the system performance and maximization of the resource usage efficiency, policies have to be dynamic, in order to effectively adapt to the continuous variations of the system conditions.

In a generic decision making model, spectrum allocation decisions are made based on i) the utility functions that are defined taking into consideration operation objectives dictated by the application requirements as well as the knowledge and experience derived by appropriate learning mechanisms, ii) the dynamic policies that are formally described with the use of appropriate policy description languages and that depend on different parameters, such as regulation, and iii) the analysis of the context information that is the result of the observation of the radio environment. The reconfigurable CR platform, with its implementation issues and hardware constraints, is responsible for the instantiation of the decisions made, influencing the radio environment and, consequently, the observed context information.

This deliverable aims at providing a summary of the scope, functions and basic mechanisms of the ACROPOLIS decision making framework. Its main focus is on policy based decision making, emphasizing on the architecture, features and characteristics of the policy frameworks with respect to CR networking. It also provides guidelines for the use and implementation of policies in CR systems, highlighting the importance of decision making. Special attention is paid on the requirements and the constraints of the implementation of a policy framework on a hardware instantiation platform.

The deliverable is organized as follows:

Section 2 begins with an overview of the existing and most notable policy frameworks for CR networks, summarizing the main architectural components of each framework and providing an insight of their merits and usefulness in CR networks. The section continues with a summary of the most well-known policy languages for CR systems, describing their basic structures and providing exemplary policy rules. Also, it summarizes the requirements for policy languages that are provided by the recently introduced IEEE P1900.5 Standard. Finally, it briefly describes the ACROPOLIS approach regarding the expression of policies to be used in the decision making framework to be developed.

Section 3 first provides an overview of the IEEE P1900.4 Standard that aims to improve the overall composite capacity and Quality of Service (QoS) of wireless systems in multiple radio access technology environments by defining an appropriate system architecture and the protocols required to facilitate the optimization of radio resource usage. This section describes its system architecture, the policy and decision making process, and the information flow between the different system entities. Then, it continues with an overview of the IEEE P1900.5 Standard, providing the general architecture requirements for the policy-based control of DSA radio systems, followed by the description of its architectural components and interfaces.

Section 4 discusses the ACROPOLIS notion of a generic policy framework. Additionally it argues on how policies can be used, i.e., deployed, in a CR system while focusing on the decision making process. Its aim is to describe a simpler, use-case independent, and more synergic policy framework capable of providing a more unified and “easy to implement” system architecture.

Section 5 describes the general requirements and constraints of a CR instantiation platform and elaborates on the applicability of the USRP2 platform, which is one of the most popular and widely used Software Defined Radio (SDR) platforms, on policy based CR scenarios, discussing on its features, hardware constraints, and policy implementation issues.

Finally, Section 6 provides a summary and some concluding remarks of this deliverable.

2. Policy frameworks and languages in Cognitive Radio Systems

This section first provides an overview of the existing and most notable policy frameworks for CR networks, namely the ARAGORN, E2R, E3, ORACLE, and IEEE 802.22 WRAN frameworks. The main architectural components of each framework are summarized, giving an understanding of their merits and usefulness in CR networks. Then, it continues with the review of the most well-known policy languages for CR systems, namely XGPL, ORACLE PL and CoRaL, describing their basic structures and providing exemplary policy rules. Also, it provides a short description of the requirements for policy languages that are provided by the recently introduced IEEE P1900.5 Standard. Finally, it briefly describes the ACROPOLIS approach regarding the expression of policies to be used in the decision making framework to be developed.

2.1 Existing policy frameworks

Policy based CRs have recently attracted increased interest by a number of research institutes, projects and standards. This subsection provides an overview of the existing and most notable policy frameworks that target the CR paradigm today.

2.1.1 The ARAGORN framework

The ARAGORN policy architecture is one of the most comprehensive operable policy architectures specifically tailored for CR environments today [1], [2]. It allows flexible CR behaviour by fostering simple and efficient policy exchanges, policy based reasoning and policy enforcement. The architecture embraces policies coming from all relevant stakeholders (i.e., regulators, operators and/or users) offering options for each of them to express their specific goals. As a result, the architecture can govern dynamic resource management through dynamic policy changes that reflect the different behaviour of various network entities.

2.1.1.1 Overview

The ARAGORN policy architecture comprises three main components, as shown in Figure 2-1:

- Policy Server (PS),
- Policy Engine (PE) and
- Cognitive Resource Manager (CRM).

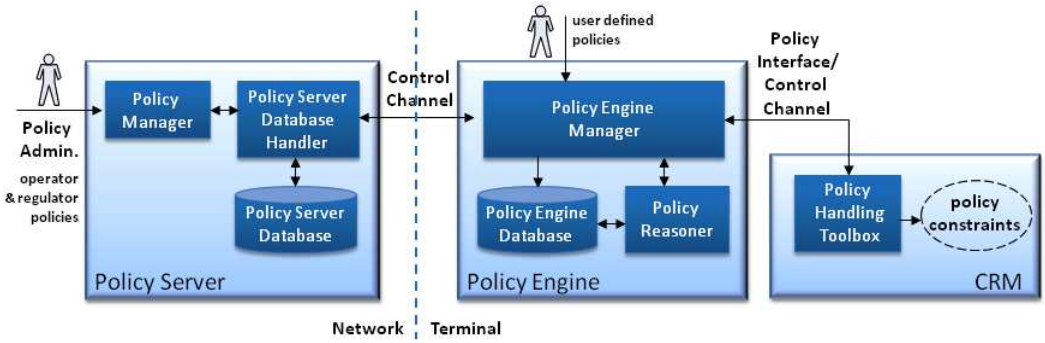

Figure 2-1: ARAGORN policy architecture [1]

The PS is the central repository for storing network policies derived by regulators and operators and for storing the information about the active users in the network. Each CR network user registers to the PS upon activation announcing its user class and device type. As a result, a specific set of policies for the respective user class and device type combination is sent to the referred user.

The PS comprises three elements: a *Policy Manager* (PM), a *Policy Server Database* (PSD) and a *Policy Server Database Handler* (PSDH). The PM is in charge of dynamic policy derivation in the network based on dynamic environment information or the administrators' input. Furthermore, this component is in charge of the user classification management. The PSD is the actual repository for policy user information data, while the PSDH is the controller of the PSD. It registers the users, stores the policies coming from the PM and distributes policies to the active network users.

The PE is a terminal's policy component, which performs the main policy task on the terminal side. It reasons based on the policy queries coming from the CRM and the active policy set dedicated to the specific terminal (user). The PE consists of three components: a *Policy Engine Manager* (PEM), a *Policy Engine Database* (PED) and a *Policy Reasoner* (PR). The PEM is the management entity of the PE that registers the user to the PS, receives and stores policies in the local PED and relays policy queries and replies to and from the PR. The PED is the local storage of the dedicated terminal (user) policies coming from the PS and the locally derived user policies. The PR is the policy element performing the reasoning process. The policy reasoning output sent to the CRM can be one of the three possible replies:

- The request is allowed
- The request is not allowed
- The request will be allowed if ... is satisfied

The ARAGORN policy architecture provides the option for remote reasoning if the terminal is not equipped with a PE.

The CRM is the main optimization, learning and decision making component in the terminal, creating and sending policy queries based on environment information and previous experience and, afterwards, making the optimization and decision making within the received policy constraints. The Policy Handling Toolbox (PHT) is the policy related CRM component adapting the policy queries in appropriate policy language format, in compliance with the PR and presenting the policy replies in form of policy constraints to the CRM.

The ARAGORN policy architecture also comprises two main interfaces: a *Policy Interface* (PI) and a *Control Channel* (CC). The PI handles the exchange of policy queries and replies between the CRM and the PE. Furthermore, this interface supports the policy change triggering and emergency situations triggering to and from the CRM. In the case of remote reasoning (the terminal does not possess a PE), the PI specific interactions are handled by the CC. The CC is the interface supporting the policy distribution and user registration between the PS and the PE. This interface allows emergency triggering and policy changes reporting options, as well.

The messages exchanged over the CC and the PI interfaces are specified by a custom policy protocol with eight messages: PolicyMsg, PolicyQuery, PolicyReply, PolicyChangeMsg, EmergencyTrigger, UserRegisterMsg, NewPolicyMsg and UsersInfoMsg. Each message has a specific purpose in compliance with the previous elaborations.

The practical realization of the ARAGORN policy architecture utilizes the neXt Generation (XG) Prolog PR [3] and CoRaL [5] in the PE due to their flexibility and extensibility. Both are modified and extended for ARAGORN policy-specific purposes. The XG Prolog PR is significantly improved by the provided support of “why not permitted?” response from the reasoning process. This enhancement highly improves the conformance checking process, minimizing the time required to converge to a permitted solution. The CoRaL built-in ontologies along with *additional ontology extensions* enable the following parameters for policy specification: frequency, bandwidth, radiated power, Radio Access Technology (RAT), application types and priorities, time and location related parameters, etc.

The organization of the policy system provides several distinct system functionalities:

- Timely distribution of policies from the central PSs;
- Efficient policy reasoning in the terminal before starting the transmission, and possibilities for remote reasoning;
- Dynamic reaction of the terminals to environmental/policies changes accommodating to the imminent conditions;
- User/terminal classification, effectuated in dedicating different set of policies for specific users group;
- Possibilities for dynamic changes of user class/device type by the policy administrator when needed;
- Options for restricting resources and applications in different time intervals;
- User policies support giving options for the users to apply their preferences;
- Emergency triggering support to notify the PS for situations when certain network resources are no longer available;
- Possibilities for straightforward extension and adaptation of the architecture due to its modular structure.

2.1.1.2 Scope and applications

This section elaborates some prominent ARAGORN policy architecture applications.

Spectrum mobility application

A spectrum mobility application of the ARAGORN policy architecture is evaluated in [2]. A central PS is controlling the access and usage of the spectrum of two USRP2 [6] based CRs in the 2.4GHz Industrial Scientific and Medical (ISM) band. The example on Figure 2-2 assumes a scenario case where, in the beginning, the USRP2s communicate on channel 1. After a certain period of time, the communication between the two CRs is changed to channel 3 (spectrum handover) due to variations of the channel conditions. The spectrum handover is fostered by a dynamic derivation and change of policies in the PS (Figure 2-2).

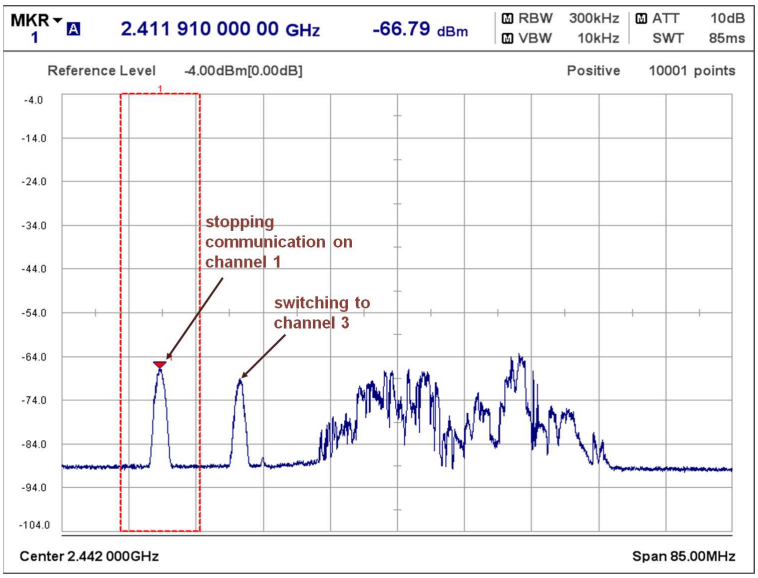

Figure 2-2: Stopping communication on Wi-Fi channel 1 (2412MHz) and starting on channel 3 (2422MHz) [1]

Additionally, the same scenario case can be used to evaluate the reactivity of the policy system to policy and environment changes. Figure 2-3 depicts the throughput of a Real-time Transport Protocol/User Datagram Protocol (RTP/UDP) streaming application while performing channel switching. The inter-channel handover duration is around 1.5s taking into consideration all the actions performed during the actual channel switching (e.g., spectrum sensing of the USRP2, message exchanges between USRP2s, actual communication between USRP2s, etc.). The actual policy messages exchange and policy reasoning require around 200ms of this time. It should be noted that the system reaction time does not depend only on the reaction times of the policy components. It also depends on the actual hardware used in the implementation (USRP2 in this case), as well as the selected communication recovery mechanism.

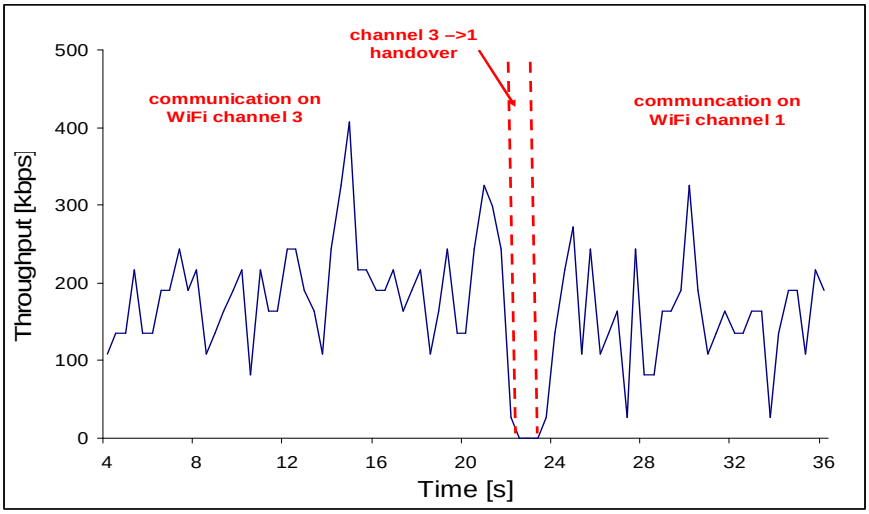

Figure 2-3: RTP over UDP streaming throughput [1]

The results presented in this subsection prove that the practical implementation of the ARAGORN policy system offers high reactivity to policy and environment changes, thus enabling real-time and efficient spectrum mobility.

Spectrum sharing application

Another potential application of the ARAGORN policy architecture is spectrum sharing [7], [8] between primary and secondary CRs or among secondary CRs only (Figure 2-4). The PM is enabled with spectrum sensing functionalities to evaluate the spectrum underutilization, detect spectrum opportunities and, thus, create sharing policies to be used by the secondary radios.

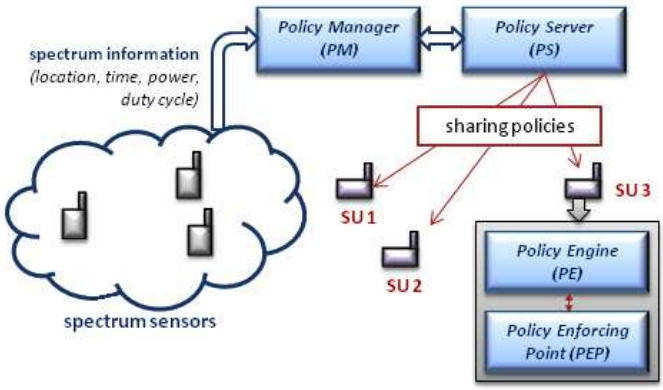

Figure 2-4: ARAGORN policy system spectrum sharing application [8]

The spectrum sharing application targets two types of policy based spectrum sharing management: proactive and reactive. The proactive management is represented by static policies. These policies are derived as a result of long term measurements of the frequency band of interest, up to several days. An example of a static CoRaL policy is the following policy *staticSharing*, which allows the secondary sharing of frequency bands in the range

[2426-2448MHz] due to its underutilization in the time period [17:00-08:00h]. The *staticSharing* policy also specifies power, bandwidth and access restrictions in order to protect the primary users.

```
Policy staticSharing is
use request_params;
defconstallowedPeriod : TimePeriod;
startTime(allowedPeriod,"T17:00:00");
endTime(allowedPeriod,"T08:00:00");
allow if
distance(onLocation(req_transmission),loc1) =< 10 and //10m from the loc1
inTimePeriod(onTime(req_transmission), allowedPeriod) and
centerFrequency(req_transmission) in {2427..2447} and //in MHz
meanEIRP(req_transmission) =<-20 and //in dBm
bandwidth(req_transmission) =< 2.5 and //in MHz
macType(req_datalink) == csmaca and
backoff(req_datalink) >= 10 //in ms
End
```

Figure 2-5: The *staticSharing* policy

The reactive policy sharing management is represented by dynamic sharing policies. These policies are evaluated focusing on the medium and short term spectrum occupancy. They are derived on-the-fly by the PM considering the spectrum occupancy in the last few minutes up to several hours of interest to cope with sudden changes to the spectrum occupancy by the primary system. Furthermore, there is a two-fold classification of the CRs into priority and non-priority radios. The priority radios are allocated to the more static frequency bands in terms of the utilization in the recent several hours, while the non-priority radios are allocated to the more critical bands (currently free, but utilized in the recent history).

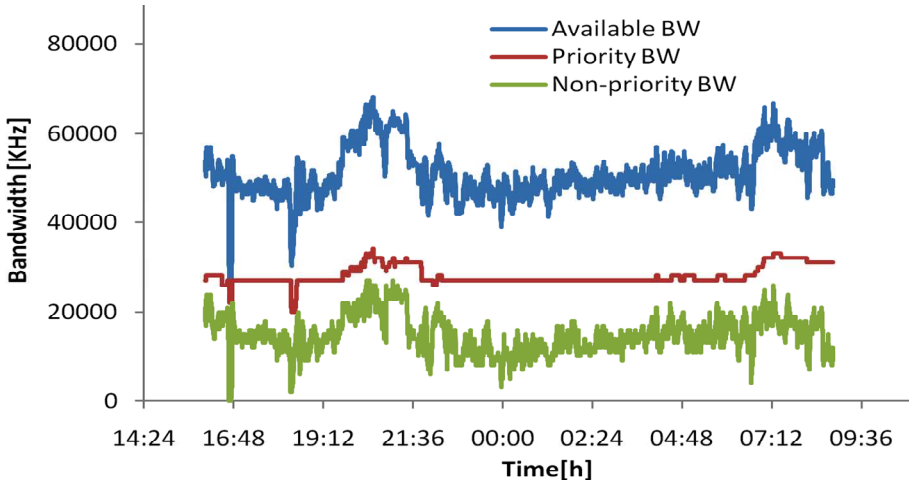

Figure 2-6: Reactive policy based spectrum sharing application – available vs. assigned bandwidth to priority and non-priority users [8]

The potentials of the spectrum sharing scheme are practically tested on a testbed in real environment conditions with focus on the channel assignment process. The targeted band is

the 2.4 GHz ISM band where the Wireless Local Area Network (WLAN) users are treated as primary. The tests were performed in a city centre flat in Skopje where the referred ISM band is overcrowded, i.e., many Access Points (APs) covering the full band are present. A laptop computer, besides the role of policy storage (PS+PM), is also enabled with sensing capabilities through attached Texas Instruments (TI) eZ430 CC2500 based spectrum sensors [9] performing energy detection and feeding the policy derivation process. The derived policies can refer to different Secondary User's (SU) classes in terms of requested bandwidth and spectrum access priority. However, the actual SU types and technologies are not a subject of interest for the presented spectrum sharing scheme as it only deals with the policy derivation process. Figure 2-6 represents practical results in the 2.4GHz ISM band plotting the available bandwidth, the allocated priority bandwidth and non-priority bandwidth. These results prove that the strategy to assign priority radios to more static bandwidth (low short-term utilization) results in higher reliability – lower number of channel repossessions by the "primary system".

2.1.2 The End-to-End Reconfigurability (E2R) and the End-to-End Efficiency (E3) Framework

The policy framework that is presented hereafter was initiated in the E2R project [10] and was enhanced during the E3 project [11]. Moreover, it is also captured, up to an extent, by the IEEE P1900.4 Standard which was approved by the Institute of Electrical and Electronics Engineers (IEEE) Standards Board on January 30th 2009 [12], [13]. Due to their close correlation, much information related to the frameworks of both E2R and E3 can be found in the sub-sections which describe P1900.4 (Section 3.1) of this document.

2.1.2.1 Overview

Overall, in both frameworks, a *Network Reconfiguration Manager* (NRM), residing on the network side, derives radio resource selection policies for managing the behaviour of terminals, which are either CRs or multimode devices operating in heterogeneous wireless environments. A *Terminal Reconfiguration Manager* (TRM) resides in each terminal and is responsible for receiving and implementing these policies, taking into account also local terminal strategies.

Moreover, a policy is formulated as a set of a Compound Policy Condition and a Policy configuration. A Compound Policy Condition consists of a Logical Expression (e.g., AND, OR, XOR) and one or more Compound Policy Conditions or Policy Conditions. A Policy Condition comprises a Policy Expression (e.g., *equals*, *greater than*, *greater equal*, and *less than*, *less equal*, etc.) and a Policy Argument. A Policy Argument includes parameters such User Class, Location, and Time zone information. Information comprised in the Policy Argument indicates the devices that are affected by the specific policy. A Policy Configuration indicates the RATs that can be operated by transceivers of Access Points/Base Stations (APs/BSs), as well as certain frequency bands per RAT in a certain service area. Moreover it also may specify the services and corresponding QoS levels that can be provided over certain RATs.

2.1.2.2 Scope and applications

Policy is a definite goal, course or method of action to guide and determine present and future decisions. In this sense, *Policies* are implemented or executed within a particular context (such as policies defined within a business unit).

A number of different types of policies are incorporated within the scope of these frameworks. Such policies are derived from the involved actors, strategies and objectives (i.e., the Network Operator, the User, etc.) and target the system entities behaviour in a specific technical area (i.e., Flexible Spectrum Management, RAT Selection, etc.). The different types of policies that have been identified are the following:

- Dynamic Spectrum Access (DSA) Policy: is derived by the operator taking into account corresponding regulatory rules for establishing the frequency ranges of the RATs.
- Radio Resource Assignment (RRA) Policy: is derived by the network and conveyed to the terminals; such policy gives straight indication to user terminals regarding radio resource assignment.
- Mobile Terminal Assignment (MTA) Policy: is derived by the network and communicated to the network elements (infrastructure manager, Base Station) for optimal terminal distribution between different Radio Access Networks (RANs) targeting resource usage optimization.
- RAT Selection Policy: is derived by the network and communicated to the user terminals for the access selection procedure.
- Energy Saving Policy: is derived by the network and conveyed to infrastructure management in order to drive the network elements behaviour for energy saving. For example, decision about switching on/off cells.
- Handover Policy: is derived and conveyed from the network to the terminal in order to assist the terminal decision regarding an inter-RAT handover.
- Self Organizing Network (SON) Policy: is derived and exchanged among network elements and user terminals in order to define the most efficient organization structure and its behaviour.

Acquiring and maintaining policy information

This section provides the results with respect to the performance of the functionality, initiated in E2R project and finalized in E3 project, for acquiring information on policies of various relevant entities (such as the network operator, local regulatory body, etc.). This functionality is mapped to mobile stations and assumes the existence of either some cognition provision entity in the form of a cognitive pilot channel/server (CPC/CPS) or direct interfaces with network support entities. It targets at identifying that subset of policies that apply within the specific context. These policies are then given as inputs during the decision process for selecting the most appropriate configurations.

The results that are presented hereafter refer to the performance evaluation of this functionality with respect to:

- Required Time for deriving policy information:
  This refers to the time that is required in order to derive policy information. Figure 2-7 presents the measured time for every policy information acquisition process in the

duration of one hour, i.e., 3600 steps of 1s each. The average time observed is approximately 28000 μs.

- Energy consumption in deriving policy information:
  This refers to the energy consumed, in terms of battery power, while deriving policy information. Figure 2-8 presents the battery level during the policy acquisition process in the duration of one hour. At the beginning of the simulation the battery was fully charged while in the end the measurements show that the battery level is around 85%.
- Central Processing Unit (CPU) usage:
  This refers to the CPU required to process and store policy information. The evolution of the CPU usage on the device during the policy acquisition process can be observed in Figure 2-9.

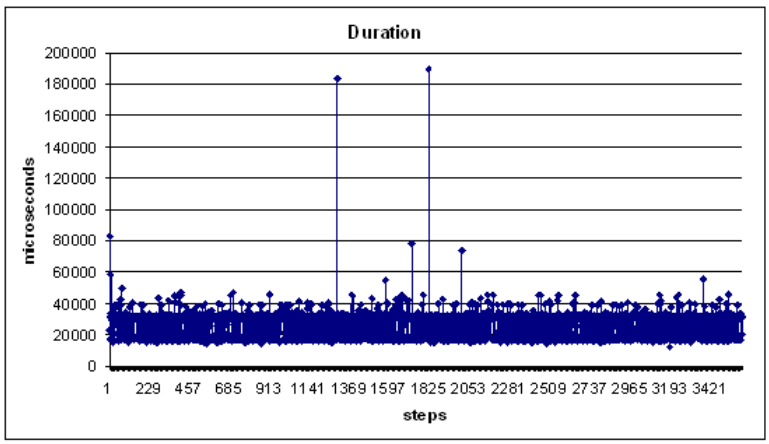

Figure 2-7: Required time (in μs) for deriving policy information [17]

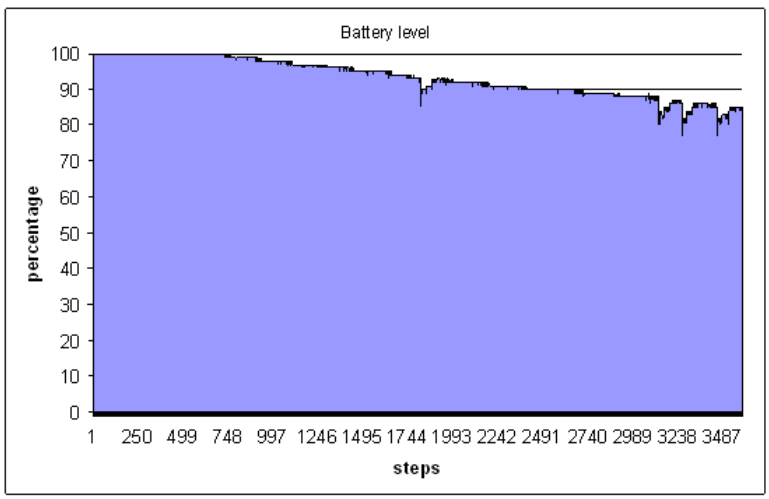

Figure 2-8: Battery level in order to acquire policies [17]

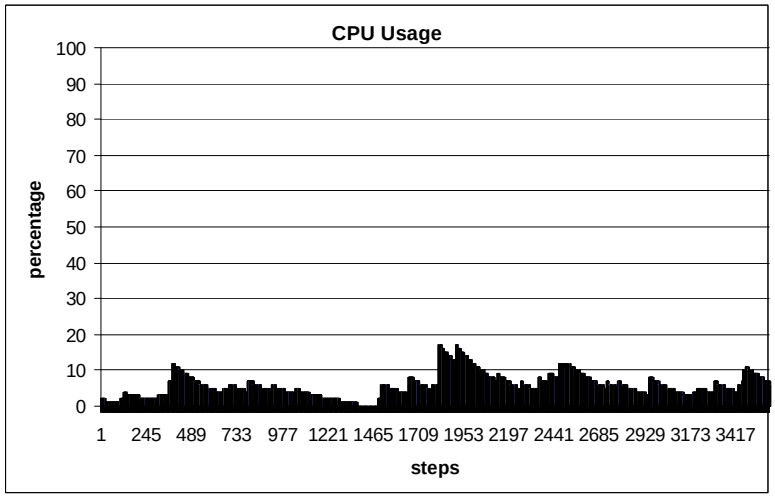

Figure 2-9: Percentage of CPU usage in order to acquire policies [17]

The overall performance of the functionality, given the above presented results, suggests that this functionality is feasible and efficient in terms of required time for deriving policy information, battery and CPU usage without imposing significant processing delays on the device.

More details with respect to this functionality and the context under which the performance evaluation took place can be found in [14]-[18].

2.1.3 The ORACLE framework

In [19], [20] and [21] the policy framework for opportunistic communication nodes is described as an output of investigations conducted within the European 6th Framework Program (FP6) project "Opportunistic Radio Communications in unLicensed Environments" (ORACLE) [22]. In the context of ORACLE, a policy is seen as a collection of rules defining various operational aspects (i.e., spectrum assignments, power levels and transmission technology). A policy framework defines a support structure (creation, storage and processing mechanisms, as well as tools for management and deployment) for operations on these polices. For flexibility, in ORACLE a rule-based reasoning engine was chosen for processing policies. This means that complex tasks of opportunity detection and decision making can be split into mutually independent operations to be processed in consecutive steps or in parallel. Applied rules are assumed to be widely configurable at run-time depending on the current context (information characterizing the situation of an entity). In ORACLE, the context data is a universal set of variables or parameters which are then used to limit the options of how a system can be used. In order to summarize and structure information for better and more efficient handling, profiles – configuration setting, which describe information associated with each entity or application(s) – are also introduced.

2.1.3.1 Overview

The architecture of the ORACLE policy framework is described in the form of high level block diagrams and data flow diagrams. The core processing engines of this architecture are:

- Raw Context Processing (RCP),
- Context Watcher (CW) and
- Reasoning Engine (REng).

In Figure 2-10 the basic information flow between those entities is shown.

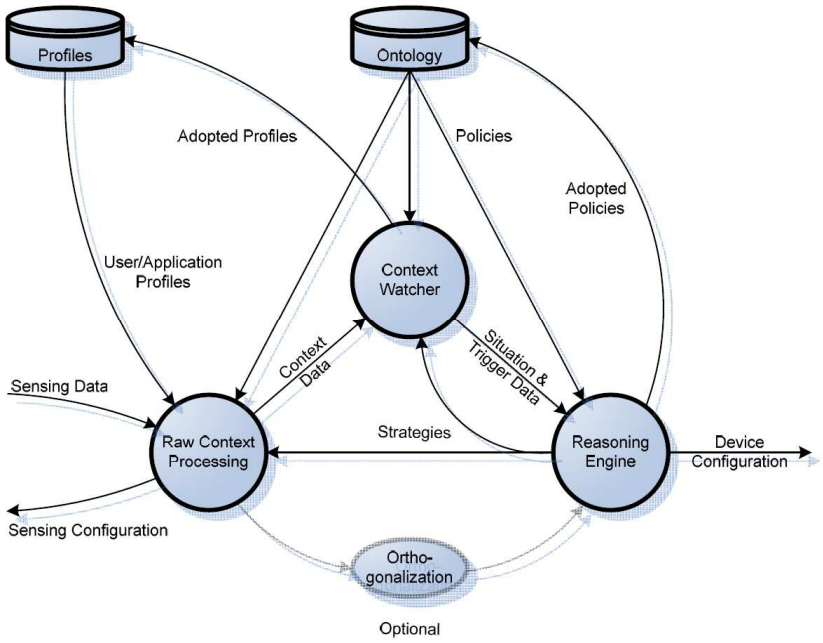

Figure 2-10: Basic Information flow in context and policy processing within in the ORACLE Policy Framework [20]

The RCP is an entity which obtains sensing data, processes them and generates the output in the form of a unified context vector. Data can be obtained from any available sensor devices or protocol layer service including both lower layer services as well as network layer (e.g., traffic flow information) or application layer (e.g., application behaviour) sensing and metering services. The RCP configures local sensing devices and may control, depending on applied strategies and rules, e.g., source, frequency, resolution and accuracy of sensing data acquisition within the limits of configurability and sensing capabilities of the terminal. The of the RCP is controlled by policy rules which are provided by the ontology and by strategies obtained from the REng as a result of earlier policy processing.

The CW is an entity observing the context parameter vectors obtained from the RCP in order to detect changes in that data. Thus, the CW generates as an output a selector dataset and a set of context parameters (called situation data in order to distinguish this pre-processed context data from raw context data), which describe the current environment model parameters and identify the changes observed. Rule-based logic determines which change should trigger policy processing by the REng. Rules and strategies to apply in CW's processing are determined by policy rules processed by the REng. It is notable that the CW may also generate new or updated existing profiles, e.g., from observed application or user behaviour.

The REng acts whenever the CW detects a context change. The main function of the REng is to match a policy rule for the opportunity specified by the current situation. In other words,

the REng should select the appropriate policy set based on its knowledge about the current and previous context. Furthermore, this entity is capable of creating or updating policies within the ontology. These might be policies received from remote terminals in the course of collaborative processing as well as locally generated policies as an outcome of the learning or self-configuration process. As an output, the REng generates a device configuration data vector and strategy selector data sets which are sent to the Configuration and Control entity. The way how the configuration data generated by the REng reflects to a physical device configuration depends only on the implementation of the device configuration primitives.

During the process of developing an ORACLE architectural model, it has been found that the above described main components are linked more tightly than expected initially and might be implemented each as a dedicated optimized version of the same rule-based inference engine. Such an approach facilitates the use of a central ontology providing both rule sets and profiles by a single database-like architectural component.

In ORACLE various data formats were defined for different interfaces between components of the policy framework. In [20] Defense Advanced Research Projects Agency (DARPA) XG [23] policy rules were identified as a basis for the initial implementation of the ORACLE policy framework. XGPL allows to link opportunities and configurations in a formal Extensible Markup Language (XML)-based description. An XG policy rule may also consider basic conditions for use (i.e., facts referring to a certain context) and can provide precedence information to support decision making. The ORACLE approach utilizes XG policy rules to map from a target opportunity to a terminal and sensor configuration.

2.1.3.2 Scope and applications

This section elaborates some prominent ORACLE policy architecture applications which are evaluated in [24]. The ORACLE showcase platform is developed based on the Ettus USRP hardware (including USRP motherboard, the RFX2400 transceiver daughterboard, Altera Cyclone Field-Programmable Gate Array (FPGA), Analog-to-Digital Converters (ADCs), Digital-to-Analog Converters (DACs) and antennas) and the GNU Radio open-source software. Ettus USRP hardware is utilized as Radio Frequency (RF) front end of the ORACLE terminal and the GNU Radio as a software interface between the Generic Algorithm-based decision making engine and the RF front end. It is notable that in order to avoid severe complications during the implementation, the decision making engine is run under MatLab environment and communicated with GNU Radio using the Python as software interface in real-time.

RF Scanner in the 2.4GHz ISM Band

The first scenario considers the implementation of a simple RF scanner based on energy detection technique. Due to limitations of the USRP hardware, the sensing process could not be completed in one scanning session by one RF front end. Thus, an adaptive frequency hopping is performed to complete the sensing of the entire 2.4 GHz band. The information gathered is recorded and processed to detect the spectrum opportunity. Figure 2-11 shows the plot of the scanning information when some terminal is transmitting data packets at the centre frequency of 2.442 GHz (channel 7 in 802.11b). The ORACLE terminal performing

spectrum sensing can pick up a strong signal and the spectrum opportunities can then be identified based on the preset energy threshold.

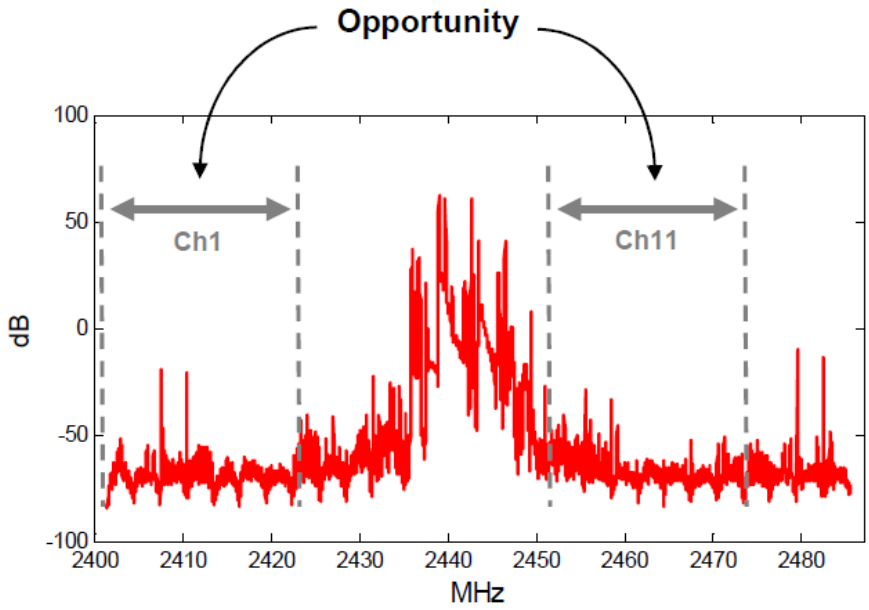

Figure 2-11: Sensing information for the 2.4 GHz band achieved by the ORACLE terminal while the other RF frontend transmits data at centre frequency of 2.442 GHz [24]

In this particular situation, channel 1 or channel 11 of the IEEE 802.11b can be used by the ORACLE terminals within the same area and thus avoid interference to the current user, which in real environment could be a Primary User or a non-collaborating user.

Decision Making for Opportunistic Channel Allocation

The main goal of the ORACLE showcase platform is to demonstrate the opportunistic channel allocation based on the decision making engine developed within the project. To this end, an appropriate test environment is set up where one of the RF frontend is connected to the workstation that plays the role of ORACLE terminal, performing periodic spectrum sensing, decision making and data transmission. In order to cope with the processing delay, these periods are extended from realistic rates to ensure that the ORACLE terminal will be able to complete the 2.4 GHz band scanning session and demonstrate data transmission according to the decision made by the ORACLE decision making engine. Furthermore, some limitations are manually input to the database of the engine simulating the possible impact of policies and profiles.

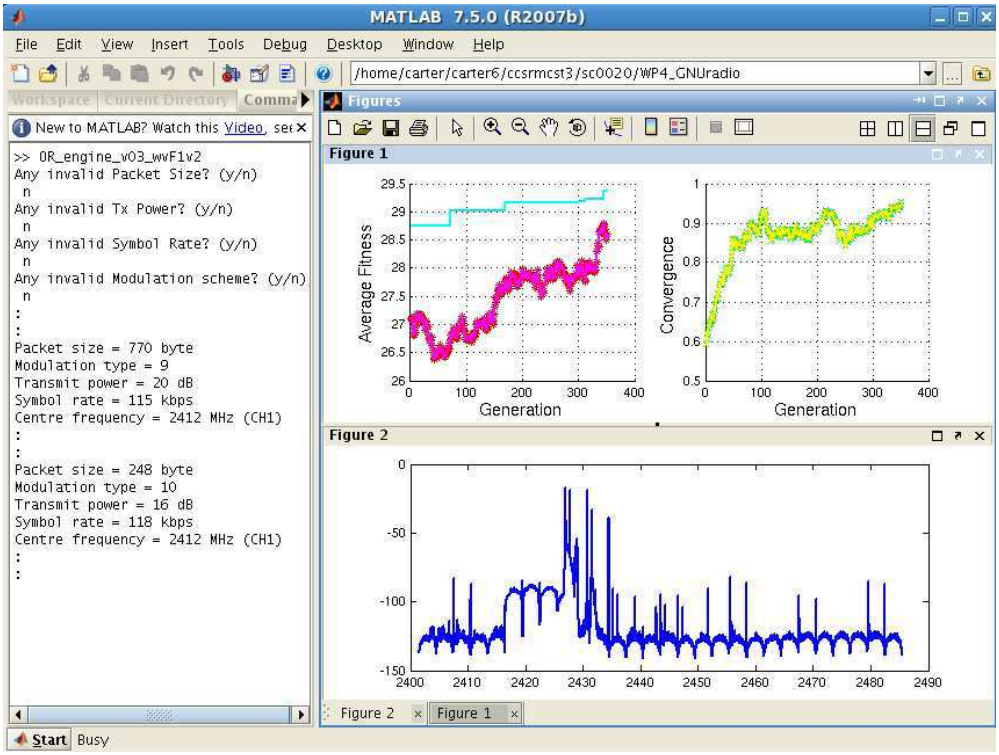

Figure 2-12: MatLab real-time display of the ORACLE decision making process [24]

Once the spectrum sensing period is completed, sensing information is collected and passed through GNU Radio interface to the decision making engine. Once the decision is made, the MatLab screen displays the controlled parameter as a result. The decision is also imported through the software interface and used to set the control parameters of the RF front end accordingly. The decision made by the ORACLE engine captured during the test can be seen in the command window on the left hand side of the MatLab screenshot presented in Figure 2-12. In this case, channel 1 in 802.11b (centre frequency = 2.412GHz) was detected as available and used for transmission. This signal can be seen on the screen of the spectrum analyser which captures the power spectrum of the entire ISM 2.4 GHz band as illustrated in Figure 2-13.

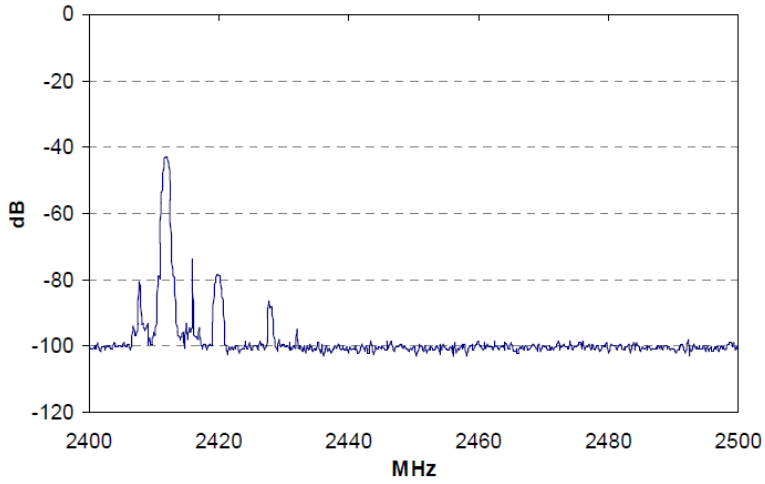

Figure 2-13: Power spectrum observed by spectrum analyser during ORACLE terminal transmission phase [24]

Since the ORACLE terminal is performing periodic spectrum sensing and data transmission, the channel selection was observed. For a short observation time, no channel switching is performed and the ORACLE terminal continued occupying channel 1 of the IEEE 802.11b.

In order to monitor the opportunistic channel allocation, another set of USRP hardware with GNU Radio is used to transmit data using the frequency channel occupied by the ORACLE terminal. Thereby, during the next sensing and decision making phase, the ORACLE terminal picked up the signal and started the process of evaluating the spectrum opportunity. As the signal transmitted by the other RF front end was detected, the decision making engine offered the new radio configurations as can be seen in the left panel of Figure 2-14.

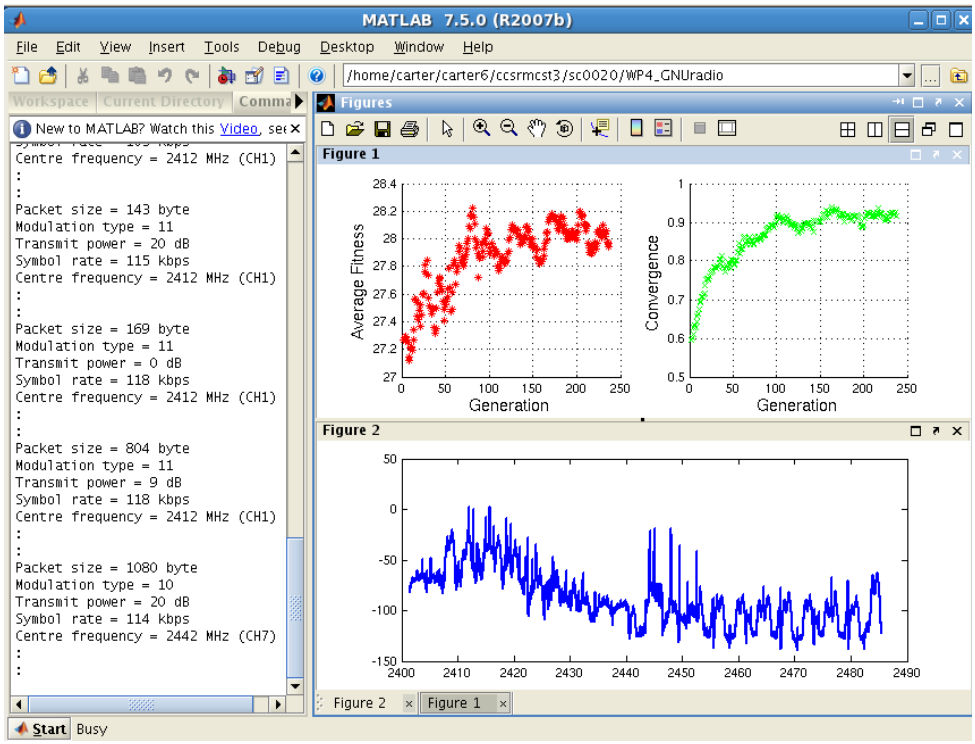

Figure 2-14: Decision made by the ORACLE engine in response to the spectrum sensing information [24]

As a result of this processing and decision made, during the next data transmission phase, the ORACLE terminal is transmitting signal at the centre frequency of 2.442GHz (channel 7 of the IEEE 802.11b). In Figure 2-15 the corresponding plot obtained from the spectrum analyser is shown. The main conclusion derived from those results is that the proper behaviour of the ORACLE decision making engine was proved in this scenario.

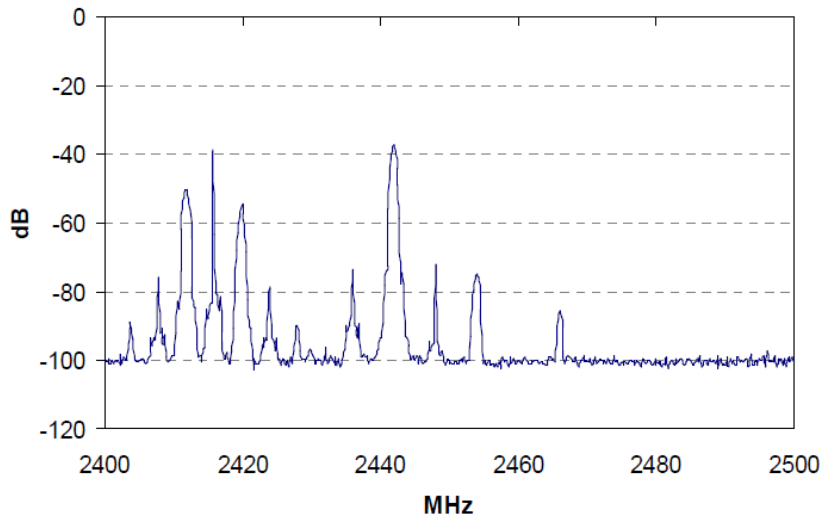

Figure 2-15: Power spectrum captured as ORACLE terminal switched to different channel while the other device continued transmitting on channel 1 [24]

2.1.4 The IEEE 802.22 WRAN framework

The IEEE 802.22 WRAN Standard [25] defines the mandatory rules that 802.22 devices should follow in order to guarantee proper protection of incumbents, compliance with regulatory domain policies, and interoperability among different WRAN implementations. To this end, IEEE 802.22 devices have to employ CR capabilities enabling them to make decisions about their behaviour in the radio system. Each decision is based on relevant information obtained a) from an appropriate database service, b) through direct spectrum sensing or c) as a result of rules governing the particular regulatory domain of interest. In the following, the IEEE 802.22 entities responsible for obtaining and managing the aforementioned information as well as their key functionalities are briefly described.

2.1.4.1 Overview

The IEEE 802.22 WRAN policy architecture comprises three main components:

- Spectrum Manager (SM),
- Spectrum Sensing Automation (SSA) and
- Spectrum Sensing Function (SSF).

The SM has to be implemented in each BS, while the SSA and the SSF functionalities should be implemented in both each BS and each customer-premises equipment (CPE) in the network. The SM is an entity responsible for the most important tasks related to obtaining, maintaining and managing information on spectrum availability as well as making and enforcing appropriate policy-compliant decisions. The SSA enables spectrum sensing as it interfaces to the SSF and executes the commands from the SM. The IEEE 802.22 Spectrum Manager and its logical interfaces are shown in Figure 2-16.

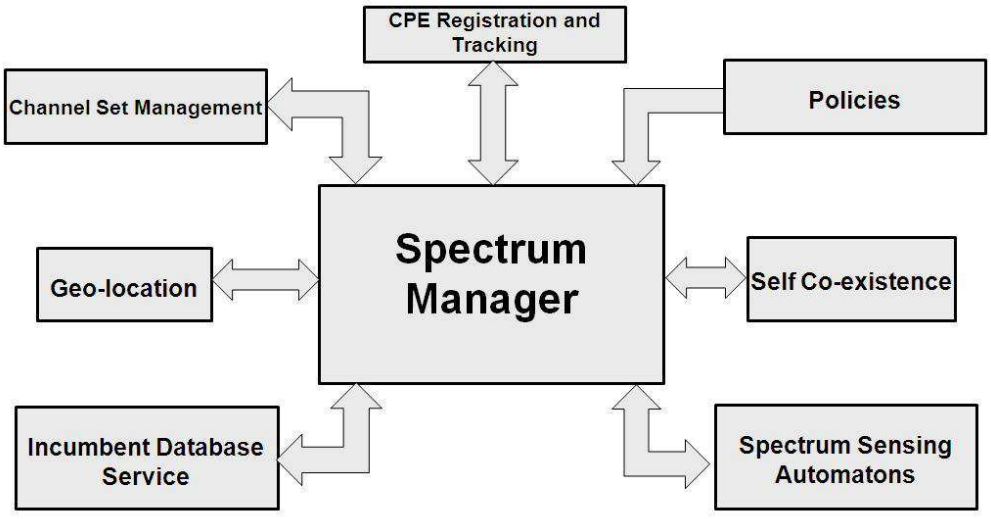

Figure 2-16: IEEE 802.22 spectrum manager and logical interfaces [25]

Spectrum Manager

The SM is the major part of the WRAN BS, which centralizes all decisions in the cell in order to ensure protection of incumbents and efficient spectrum utilization with respect to regulatory policies. The main functionalities of the SM are listed below.

- Spectrum availability information maintenance – the SM should maintain the status of the channels available for WRAN operation at its location within a regulatory domain according to the policies and rules established for that domain. Decisions should be based on information gathered from (at least) incumbent databases, geolocation service and spectrum sensing results.
- Channel classification and selection – based on available information the SM should determine operating channel which in turn will be assigned to the Physical / Medium Access Control (PHY/MAC) modules in the WRAN. Furthermore, the rest of the channels indicated by the database service as potentially available for WRAN operation should be classified by the SM using the following categories: disallowed, backup, candidate, protected and unclassified. For detailed information on the channel classification procedure (in particular, channel set transition diagram and matrix) and channel prioritization using spectrum etiquette, the reader is referred to the IEEE 802.22 Standard [25].
- Channel set management, making channel move decisions – based on the information obtained, the SM should perform perpetual control of the operating channels in order to ensure protection of incumbents and efficient spectrum utilization.
- Association control – the SM is in charge of granting or denying association rights to the requesting CPEs taking into consideration location information, basic and registered capabilities of each requesting CPE, the list of available channels and corresponding Equivalent Isotropic Radiated Power (EIRP) limitations.
- Self-coexistence with other WRANs – this functionality comprises all procedures performed by the SM (e.g., scheduling quiet periods for spectrum sensing) that facilitate proper co-operation of neighbouring WRAN BSs.
- Enforcing IEEE 802.22 and regulatory domain policies – in order to guarantee simultaneously the required protection of incumbents and QoS for the WRAN users, the SM should operate with respect to the policies specified in the IEEE 802.22 Standard. In Table 2-1 exemplary polices are provided.

Policy ID	**IEEE WRAN component involved / Event trigger**	**Event description**	**Action**
1a	BS / DBS	SM is directed by the database service (DBS) that the current operating channel is no longer available for the BS	Initiate a channel switch of the entire cell to a new operating channel – the highest priority backup channel.
1b	CPE / DBS	SM is directed by the DBS that the current operating channel is no longer	Option 1: Initiate a channel switch of the entire cell to a new operating channel – the highest priority backup channel.

		available for some of the CPEs	Option 2: Disassociate the CPEs that are not allowed to operate on the current channel and continue normal operation with the other CPEs. Optionally, the BS may signal the affected CPEs to move to a particular channel in order to re-associate with another BS and continue their operation.
1c	BS / DBS	SM obtained information from the DBS that indicates the current operating channel will become unavailable for the BS at a specific time in the future	Option 1: Initiate (after the time specified by the DBS) a channel switch of the entire cell to a new operating channel – the highest priority backup channel. Option 2: Disassociate (after the time specified by the DBS) the CPEs that are not allowed to operate on the current channel and continue normal operation with the other CPEs. Optionally, the BS may signal the affected CPEs to move to a particular channel in order to re-associate with another BS and continue their operation.
2	BS or CPE / Signal detected	The signal detected on the operating channel or either of its first adjacent channels is a Television (TV) signal through the BS spectrum sensing function or through a combination of sensing results from multiple CPEs.	If the local regulatory domain requires vacating the channel on confirmation of the presence of a TV signal, initiate a channel switch of the entire cell to a new operating channel – the highest priority backup channel.
3a	BS or CPE / Signal detected	If the signal detected on the operating channel is a wireless microphone signal through the BS spectrum sensing function, or through the sensing results from a CPE, or a combination of multiple CPEs.	Depending on additional conditions: initiate a channel switch of the entire cell to a new operating channel – the highest priority backup channel or disassociate the CPEs that are within the protected radius of the wireless microphone operation and continue normal operation with the other CPEs. Optionally, the BS may signal the next channel to go to for the disassociated CPEs before shutting down the communication.
3b	BS or CPE / Signal detected	If the signal detected on the operating channel is an IEEE 802.22.1 wireless microphone beacon signal.	Depending on additional conditions: complete cell move – initiate (preceded or not preceded by beacon authentication) a channel switch of the entire cell to a new operating channel – the highest priority backup channel specific CPEs case to operate or move – disassociates (preceded or not preceded by beacon authentication) the CPEs that are within less than *Microphone_Protection_Radius* from the wireless microphone operation and continue normal operation with other CPEs. Optionally, the BS may signal the next channel to go to for the disassociated CPEs before shutting down the communication.
4	BS or CPE / DBS or Signal detected	There is no backup channel available AND the database service indicates that the current operating channel is not available or the signal	Terminate the operation of the entire cell in the current operating channel

		detected on the operating channel is a wireless microphone or an IEEE 802.22.1 signal, or, in case of a TV signal, the signal is detected on the operating or any of its first adjacent channels	
8	CPE / Geolocation	BS has determined that the position of the CPE has changed by greater than that specified by the local regulations.	The BS should request the CPE to geolocate and report its position to verify the change in location. If the location is confirmed to have changed, the BS shall immediately obtain a new list of available channels from the database service based on the new location of the CPE. The CPE shall abide by the EIRP limit specified by the database service or, if not available, abide by the regulatory requirements. If the service for the affected CPE on the current operating channel at the new location is prohibited or if the device type is fixed, then the BS shall de-register the CPE.

Table 2-1: Some of the IEEE 802.22 Spectrum Manager policies [25]

As it was stated above, the SM is a mandatory, central entity of each IEEE 802.22 BS, being responsible for the most important task related to obtaining and enforcing WRAN and regulatory domain policies. Thus, all possible operations of the SM are precisely described in the IEEE 802.22 Standard. Allowable SM procedures are as follows: Find_Operating_Channel, Establish_Network, Registration_and_Tracking, Database_Update, Determine_Signal_Type_Execute_Policies, Background_Processes, Initiate_Channel_Move, Self_Coexistence_Mode. A diagram of the SM state machine is presented in Figure 2-17.

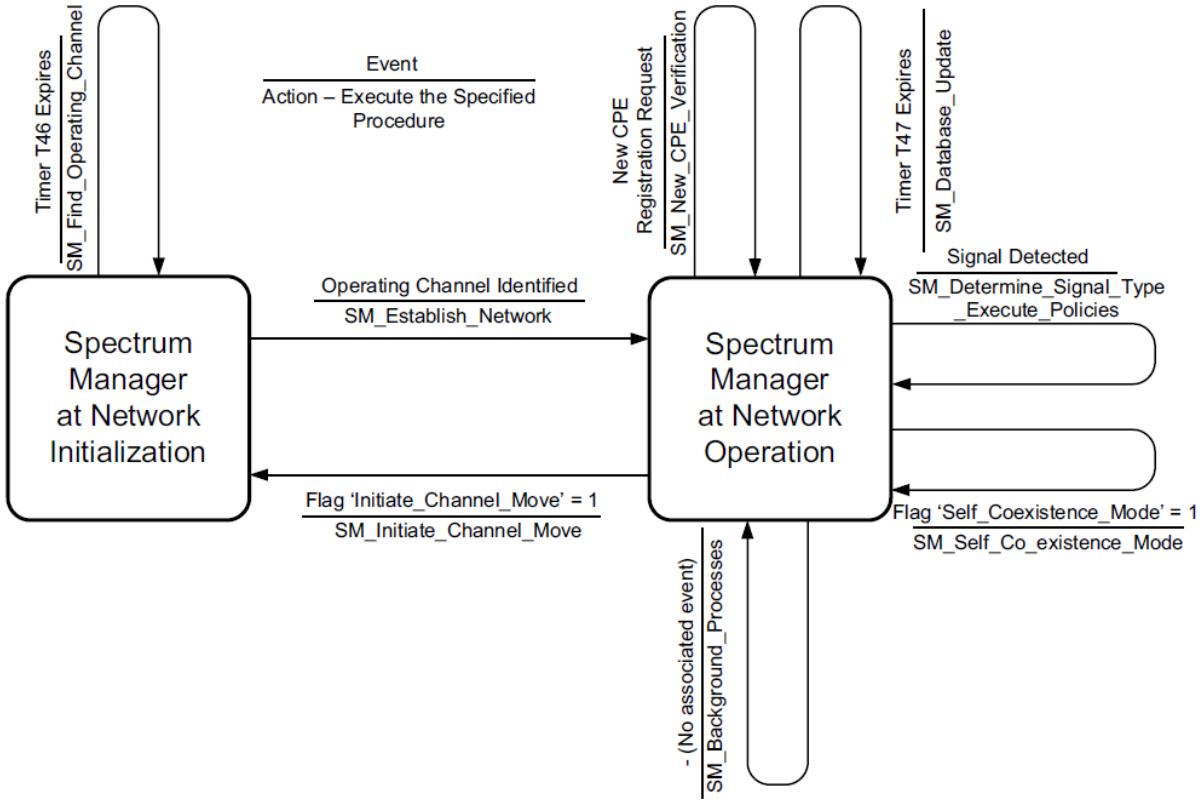

Figure 2-17: IEEE 802.22 Spectrum Manager state machine [25]

As it can be seen, the SM has two states of operation: the SM at Network Initialization, and the SM at Network Operation. The main aim of the initialization phase is to find the operating channel while during the Network Operation state, the SM should execute all its aforementioned tasks with respect to policies described in Table 2-1.

Spectrum Sensing Automation

The SSA is a mandatory entity existing in all IEEE 802.22 devices (BSs and CPEs), which enables spectrum sensing by connecting to the SSF and executing the commands from the SM. Although the SM normally controls the behaviour of the SSA, in the following conditions the SSA should control its sensing behaviour locally:

- at the initial turn-on of the BS before it starts to transmit any signal,
- at the initial turn-on of the CPE before association is established with the BS,
- during the quiet periods defined by the SM and signalled by the BS through the Superframe Control Header (SCH) for in-band sensing,
- during out-of-band sensing at the BS when it is not transmitting,
- during idle time at the CPE when the BS has not attributed any specific task to the CPE sensing signal path through the Bulk Measurement Request (BLM-REQ) message, when the CPE does not transmit or, if the WRAN operation and RF sensing use the same tuner, when the CPE does not transmit or receive,
- when the CPE loses contact with its BS.

The SSA allowable operations are defined in the IEEE 802.22 Standard in a similar way as in SM, in order to cover the functionality of the SSA in the circumstances listed above.

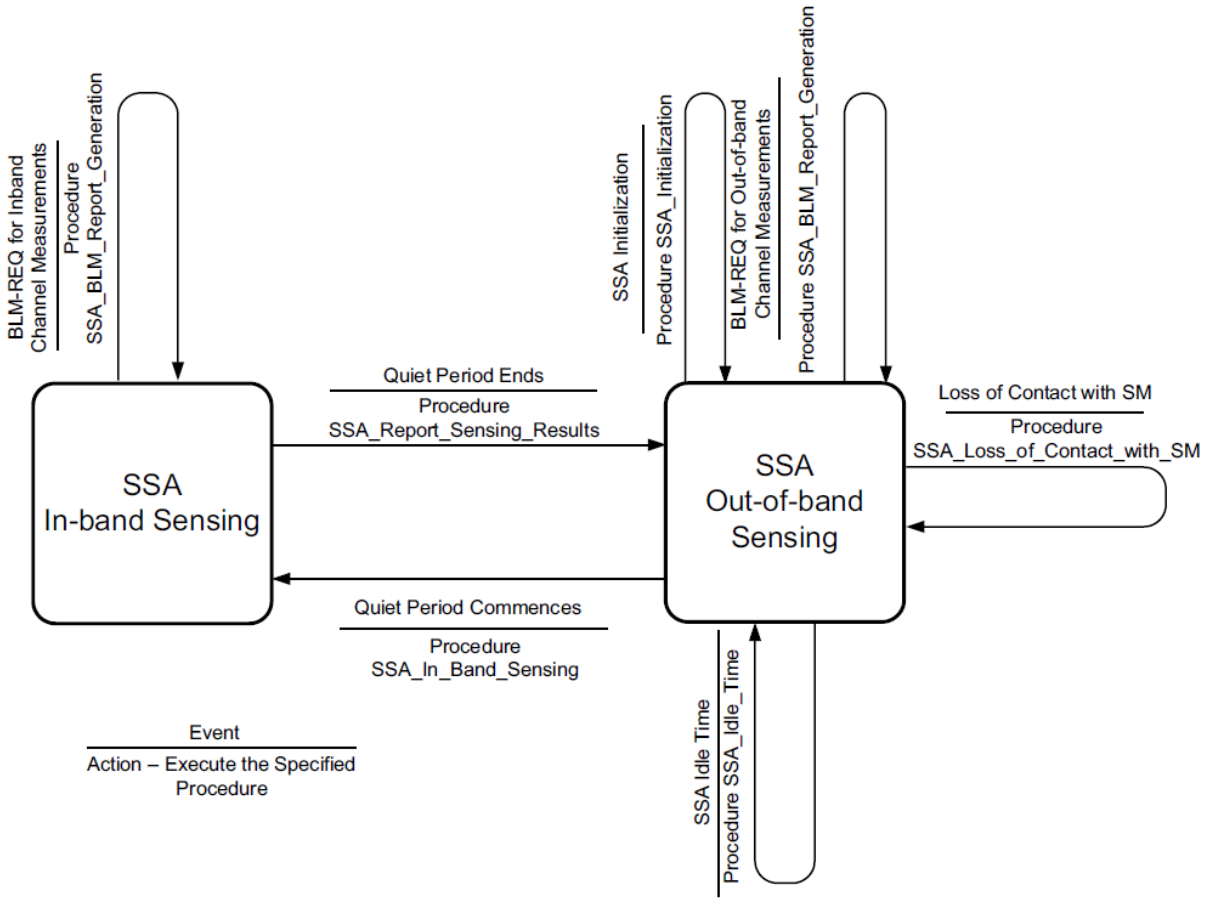

Figure 2-18: IEEE 802.22 Spectrum Sensing Automation state machine [25]

Allowable SSA procedures are as follows: Initialization, In-band_Sensing, Idle_Time – SSA operation during CPE idle time, Loss_of_Contact_with_SM, BLM_Report_Generation. The diagram of the SSA state machine is presented in Figure 2-18.

Spectrum Sensing Function

The SSF is driven by the SSA and should be implemented in each BSs and CPEs in the WRAN. The main aim of the SSF is to observe the RF spectrum of television channels and report results of these measurements to the SM via its associated SSA. Thus, the proper operation of the SSF is necessary to ensure correct triggering of SM's and SSA's actions. The IEEE 802.22 Standard allows the use of any specific spectrum sensing technique as long as its inputs, outputs and behaviour meet the specification.

2.2 Policy languages

2.2.1 Requirements of policy languages

As mentioned in the previous subsections, a policy defines the rules that can be used to control the behaviour of the nodes and manage the available resources of a network. In modern wireless systems, the policies are usually defined by local (national) or international regulations, are static and pre stored in certain devices. However, one can observe that due to the static nature of the actual spectrum allocation policies, several problems have to be dealt with, such as the widely observed spectrum scarcity and the problem of efficient deployment. On the other hand, measurements show that the licensed frequency bands are strongly underutilized. Such a problem can be solved by the application of CR networks utilizing various, declarative spectrum policies. In such an approach, the cognitive terminal has to be able to collect information about the environment (by means of sensing or connection to the geolocation databases), recognize the actual transmission needs, infer, based on the transmission policies, about the current transmission possibilities and, if the transmission is possible, about the detailed setup. Such an approach shall result in more effective spectrum management, easier deployment and verification.

DARPA's XG Communication Program [4] considers opportunistic spectrum access and proposes a novel XG Radio architecture shown in Figure 2-19. Looking at this figure, one can recognize the presence of the Strategy Reasoner module (that formulates transmission queries to the policy reasoned based on the sensed terms and current expectations regarding transmission parameters) and the Policy Reasoner (that accepts transmission requests and validate the conformance with the policies).

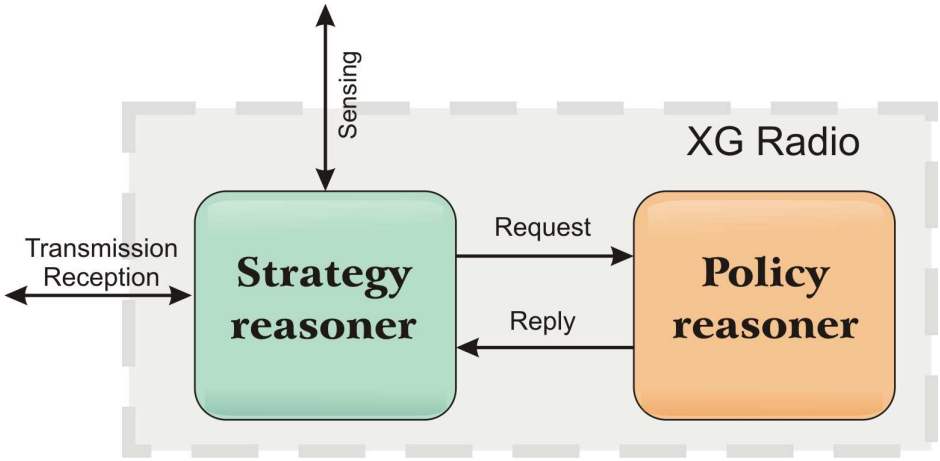

Figure 2-19: neXt Generation radio architecture proposed within DARPA XG Program

One of the main problems is to define the effective way of communication between strategy and policy reasoners (or, in a more general sense, between the module that requests the transmission and the module that validates the policy conformance). The appropriately designed policy language could be used as a solution. However, it has to be characterized by such features that allow the use of the policy language in the following situations, typical for CR:

- the actual policy can be dynamically modified and loaded to the cognitive terminals,

- the new policies can be added to the actual set of policies,
- the policy shall be decoupled from the radio implementation,
- the policy shall be extensible (i.e., we cannot predict the future requirements).

The above requirements result in the policy language being, on the one hand, very generic (it should be able to utilize the new parameters that will be defined in the future), but on the other hand very detailed (the policy reasoner has to be able to provide detailed answers to the spectral reasoner in terms of e.g., allowed EIRP values, emission masks, etc.). From the above description, one can conclude that the most appropriate solution will be to propose a *declarative* language (like Prolog, Maude, etc.), in contrast to an imperative one (such as C, C#, etc.). However, in order to use such a language its syntax, semantics and the used ontologies have to be defined. Based on [26], the following definitions can be provided:

- formal syntax – defines the rules for deciding whether a given string is in the language,
- formal semantics –defines the interpretations of the terms and expressions (words, sentences etc.),
- ontology (not in philosophical meaning) – specifies the concept of the given knowledge domain, attributes of these concepts and relations among them. It is used to infer about the entities within that domain, and may be used to describe the domain.

Moreover, from the CR point of view, the communication between the policy and spectrum reasoners ought to be adaptive, if necessary. In the simplest case, such a communication is completed in two steps: in the first step, the spectrum reasoner asks the policy reasoner for permission to transmit, and, in the second step, the policy reasoner answers *yes* or *no*. However, the policy reasoner can also provide the conditional permission, *yes but under the following constraints* or can ask for some more details, e.g., about the certain sensed parameters.

The need for a solid language for cognitive wireless systems has been expressed in [27], and the call has been answered by some international and well-recognized bodies like. End-2-End Reconfigurability, Modelling Language for Mobility within SDR Forum, SRI International, SCC41 1900.5 group (“Policy Language and Architectures for Managing Cognitive Radio for Dynamic Spectrum Access Application”).

Up to now, a couple of policy languages have been proposed that fulfil all of the aforementioned requirements. These are briefly described below. We start with one of the first policy languages devoted for CR, i.e., CoRaL, and then follow by XGPL and ORACLE PL.

2.2.2 The XGPL and ORACLE concepts

2.2.2.1 XGPL

XGPL [23] was developed as part of the U.S. DARPA XG radio development program. XGPL was the first example of a policy-based management framework proposed for CRs. The language itself (XGPL) is grounded on the principles and ideas developed and documented in Rei [31] and KaoS [32]. The XG research program developed an architecture and framework

(i.e., the XG Architecture) that was required to demonstrate the ability to efficiently utilize unused spectrum by opportunistic use. The application scenarios were limited to military deployments and, in a wider sense, to public safety use. The XGPL relies on the Web Ontology Language (OWL) in its lexical representation, based on the Resource Description Framework (RDF) and XML.

In the XG policy framework, a shorthand notation based on "C Language Integrated Production System" (CLIPS) [33] is used to document the syntax and the semantics of the language. A range of tools exist to translate back and forth between OWL and XGPL and to verify the ontology and the rule sets. These tools can also be used to identify and determine if there are any possible rule conflicts.

There are three basic constructs in the design of the XG policy language framework: facts, expressions, and rules. Policies encoded in XGPL consist of a set of facts expressed as OWL statements that describe the policy concept; there are also expressions that can be used to define a (spectrum) opportunity, a usage description, or to define membership in a policy group. Rule constructs, in general, are used to specify processing logic for policies and they have the form: "*condition-implies-action*". A policy rule consists of three elements:

- The *selector description*: filters policies to a specific environment.
- The *opportunity description*: specifies the conditions that spectrum is considered as unused.
- The *usage constraint description*: specifies the behaviour of the CR when using spectrum opportunity.

An actual policy rule in XGPL would have a format as shown in Figure 2-20:

```
( PolicyRule
( id          id )
( deny        deny )
( selDesc     selector )
( oppDesc     opportunity )
( useDesc     constraint )
)
```

Figure 2-20: XGPL policy rule format

The XGPL is object oriented (OO), relies on class definitions and uses OO features such as class inheritance. This means that the terms printed in bold letters in Figure 2-20, are not to be seen as keywords as done in the Policy Description Language (PDL) [34] or Ponder [35], but they have to be seen as class names (templates, facts, etc.) that are defined in the ontology.

- The **id** property (id: cardinal) uniquely identifies the policy rule and can be used to reference to a particular policy.
- The **deny** property (deny: Boolean) determines if a match of the selection descriptor in rule processing causes the opportunity described by the opportunity descriptor to be granted or to be denied.

- The **selDesc** (selector: policy selector) property defines a fact consisting of an authority description, a frequency description, a spatial region description, a temporal description and a device description defined in the ontology.
- The **oppDesc** (opportunity: opportunity description) property defines a fact consisting of a unique id and an expression specifying the parameters that describe the opportunity. To these parameters the radio must bind values, i.e., must provide appropriate sensing information.
- The **useDesc** (constraint: usage constraints description) property defines a fact consisting of a unique id and an expression specifying a set of parameters that define, boundaries for radio configuration parameters, e.g., giving the maximum transmit power within the frequency band selected.

The XGPL also allows the definition of meta-policies that are effective on two or more policies or policy sets (policy groups) and that describe how the set of policy rules must be processed, i.e., by grouping (providing a single reference to multiple policies), precedence (giving an evaluation order in case of conflicts), and disjunction (determining the behaviour of usage constraints in a group of policies).

2.2.2.2 ORACLE PL

The ORACLE project [22] also defines a policy language, based on the XGPL and extending its scope by making it more suitable for commercial communication scenarios in which spectrum resources are opportunistically accessed. In ORACLE, a policy statement is seen as a collection of rules. Such a set of rules can define various operational aspects, including spectrum assignment, power levels, transmission technology, etc., and these rules do this for each Opportunistic Radio (OR) terminal. In other words, policies together with the platform constraints influence (or control) the physical capabilities of the radio (e.g., power output, frequency range, modes of operation, etc.). A range of different aspects in wireless communication environments can be covered and captured in such rules. The rules themselves are derived from various sources; this includes rules issued by regulators or assigned to spectrum licenses, access guidelines or requirements demanded by operators, etc., but they can also be derived from profiles such as user profiles, or terminal/equipment profiles. In the ORACLE framework, a policy engine processes the different types of inputs and provides policies, or laws, that then again are used to define the operational profiles for OR terminals. In general, policies may be set by a variety of players. In the ORACLE cases, profiles from the following actors are considered:

- **User:** user parameters, such as QoS agreements.
- **Regulator:** this will define constraints on the spectrum such as usage in commercial and military applications.
- **Terminal:** capabilities of the terminal, multi-mode terminal capabilities, power limitations, etc.
- **Operator:** spectrum usage patterns (spectrum re-farming, opportunistic use of licensed spectrum, etc.); interference management and mitigation policies.
- **Market:** demand variation for certain services in certain locations.

Once a policy is available, it can fall into one of two categories of laws, namely the absolute laws and the relative laws. The players will set their policies in a way that they optimize their gains and they may formulate them as absolute or relative.

Policies in ORACLE are used to derive the requirements of users, terminals, network operators, regulatory bodies, etc., and together with the current system (e.g., radio environment) information, they form the OR context. In ORACLE, the term "context" describes the universal set of variables or parameters which are to be used to limit the options of how a system can be used. In the OR case, they limit the usable, or operational range of where and how spectrum/radio resources can be opportunistically exploited, see Figure 2-21.

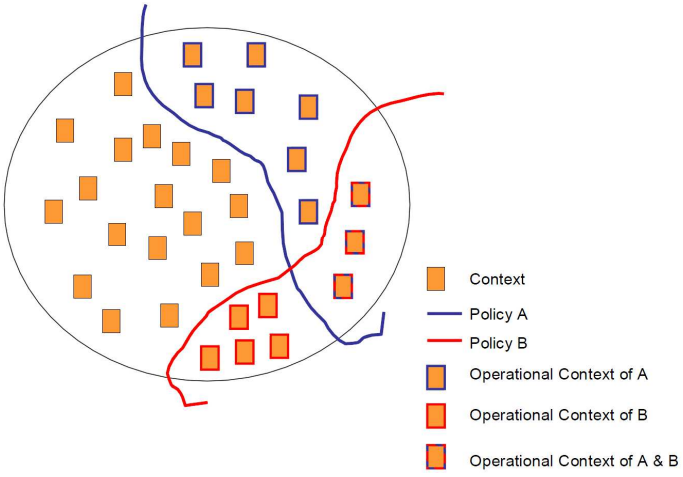

Figure 2-21: Context and application of policies [22]

Generally, the context can be seen as some function or mode of the parameters of the environment such as time, geographical location, etc. The values of these parameters are acquired by using a pre-defined set of sensors and then extracting features from these low level sensor readings. Due to the dynamic nature of these parameters and state transitions of the data sources, acquiring context information is not straightforward. Context can be extracted from low level sensors (radio context) as well as (the higher layer context information) from network nodes (access points). Context aware computing and online/offline processing are used to derive the required context information fragments (Figure 2-21).

Within the whole set of context information fragments, each policy can be used to filter and represent a subset of this universal set of context information fragments. These subsets are referred to as operational contexts. In other words, an operational context describes all the possible options that can be allowed when a policy is applied, limiting the range of possibilities (opportunities). Once all necessary policies have been applied, the decision logic evaluates the most advantageous opportunity. The decision logic uses again some policies to determine the most suitable optimization approach to make the opportunistic decision.

As indicated, ORACLE uses the XGPL as the basis of its policy framework and extends it to derive and limit the operational context of OR terminals as well as to support the decision making for spectrum access.

2.2.3 Cognitive Radio Language (CoRaL)

CoRaL has been proposed by SRI International for DARPA neXt Generation Program. As provided in [28], the language design was based on four main goals:

- to be expressive enough to express various XG radio use cases,
- to support inference/reasoning capabilities,
- to be flexible and extensible enough to be long lived,
- to support machine readability.

Let us stress that the BBN Technologies has also developed the XGPL Framework described earlier in this chapter. CoRaL is a separate language for policy specification that is different than the BBN's proposal. In particular, CoRaL is not based on OWL, although it uses the XML-based syntax. CoRaL is "a declarative language based on a typed version of classical first-order logic enriched by built-in and user-defined concepts" [29].

CoRaL supports permissive and restrictive policies, which describe conditions under which transmission is allowed or disallowed, respectively. It means that the CoRaL has to provide at least two built-in predicates, namely "allow" and "disallow". At the same time, the policy has to possess one "allow" or "disallow" rule. In general, the rules in CoRaL are presented in the generic form as follows: "allow if" conditions and "disallow if" conditions. The exemplary polices that can be encoded in CoRaL are presented below (based on [29]):

- Frequency band: Allow transmission between 670MHz and 740MHz
- Time and Location: Allow transmission between 9:00am and 3pm if the radio is at most 2km from the geographic coordinates X, Y.

In CoRaL, the ontologies and domain concepts are defined in terms of types and subtypes declarations, terms, functions, predicates, which will be shortly presented below.

Types

CoRaL has a static type system, in which a type is a set (i.e., the type *Int* represents the set of all integers, and *Int->Int* describes the whole set of all functions that maps integers to integers). Furthermore, it has some predefined, so called built-in types, such as: *Float* (floating point numbers), *Bool* (boolean values) and the aforementioned *Int*. Knowing the basic types, one can construct the following structures: [*Bool*] – is the list of booleans, {*Int*} – is the set of continuous integers, (*Int, Int*) – represents the tuples of the integer type, *Int -> Int* represents the function and *Pred*(*Int, Int)* is the type of predicates with two Integer arguments.

Terms

Terms typically represent values (are the entities that represent values [29]), like floating points (e.g., 2.87), variables (e.g., *?x*), functions (e.g., *factorial*(5)), constants (introduced by constant declarations), tuples (e.g., (2,4)), etc.

Formulas

CoRaL is an atomic language, i.e., the atomic formulas connected by the boolean connectives and quantifiers are used to define the whole statement. Three basic atomic formulas can be distinguished and defined:

- *Standard formulas* – that consist of a predicate constant and appropriate terms (assuming that we have defined the constant const p: Int, the exemplary atomic formula can be written as *p*(*17*)).
- *Constraint formula* – b<15 (other operators can be used, i.e., <,>, <=, >=, also the in operator is defined that results true if and only if the given value is on the provided list, e.g., *15 in* [1, 4, 15, 22])
- Equalities, b=15.

The following boolean connectives of typical meaning (in terms of mathematical logic) can be used: *and, or, not, implies*. For example, *p*(*x*) *or x*>=4. Two types of quantifiers exist, i.e., *exists* (existential quantification) and *forall* (universal quantification) with the standard interpretation. The exemplary use of the quantifier is provided here: (*forall ?x,?y : Float, ?z : Int in {1 .. 10}) ?x + ?y < ?z.*

Statements

The following statements are valid within CoRaL: *type declarations*, e.g., type Radio (introducing new type), *type definitions*, e.g., deftype Frequency = Float (introducing new names for the existing type), *subtype declarations*, e.g., subtype SignalDetector < Detector, *constant declarations*, e.g., const r: Int; *constant definitions*, e.g., defconst frequencies : [Frequency] = [5000, 5500, 6000];

Polices, ontologies, and requests

Policies are of the form:

policy P1 is

statements

end

Based on the previous discussion there must be at least one permissive and one restrictive rule, which are logical axioms expressing the various conditions under which the predicate holds. The exemplary policy is presented in Figure 2-22.

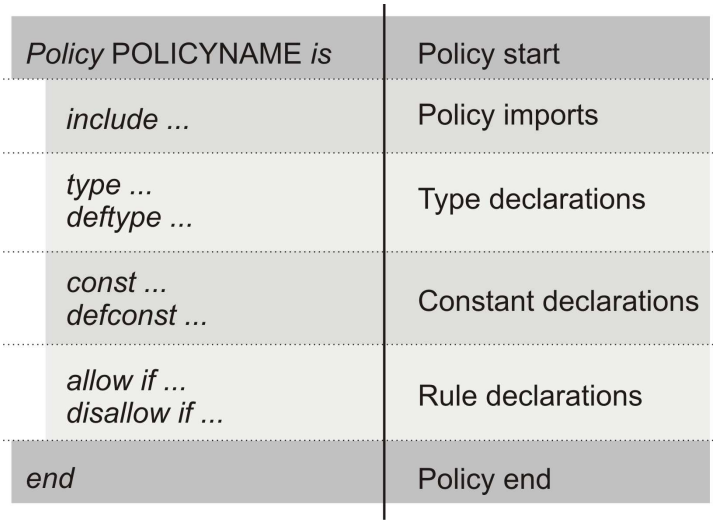

Figure 2-22: The structure of the policy

On the other hand, within the ontology any statements except allow and disallow rules are defined. It has the form very similar to the policy form, i.e.:

ontology o1 is

statements

end

The main difference between policies and ontologies lays in the fact that the policy has to contain one rule and maybe defined concepts, while ontologies only defined concepts.

Finally, as the name request suggests, a CoRaL transmission request is simply a set of constraints over the requested parameters. The general structure of the request is similar to the one of policy and ontology:

request r1 is

statements

end

Requests cannot contain statements that define type and constants, as well as the allow/disallow rules.

Examples

Following [5], we present a very simple exemplary policy expressed by means of CoRaL. Within the permissive policy POL, the transmission is allowed if and only if the requested transmit band is within the range 670MHz to 740MHz, if the radio is at most 10km away from the specific geographic point of the coordinates X, Y, and only between 9am to 3pm. The policy will be then defined as follows:

```
Policy POL is
        use req_ontology;
        allow if
                centerFrequency(req_transmission) in {670.0 .. 740} and
                (exists ?loc:LocationEvidence)
                        req_evidence(?loc) and
                        distance(location(?loc), loc1) =<10000 and
                (exists ?time_ev:TimeEvidence)
                        req_evidence(?time_ev) and
                        hour(timestamp(?time_ev)) in {9 .. 15}
end
```

In the above example, the use phrase is used for importing the *req_ontology*, that defines the requested types, constants, predicates, etc. (i.e., *req_transmission, req_evidence, LocationEvidence, TimeEvidence*). The notion *evidence* is used to represent the sensed data about the environment. *Req_transmission* is the requested transmission centre frequency that has to be within the specific range. The fourth and seventh line check if the terms *?loc* and *?time_ev* of the specific features exist.

A simple example of the request is presented below, in which the specific value of the transmission frequency is queried.

```
Request REQ1 is
        centerFrequency(req_transmission) = 670;
end
```

Ontologies and extensions

CoRaL was originally designed as a policy description language that covers the majority of DSA-related functionalities. Therefore, the in-built CoRaL ontologies define the main DSA related parameters providing means to identify the spatio-temporal-frequency solution space for the DSA-enabled terminal.

The ontology *time* defines the main time related types, such as:

- TimeInstant – defined with year, month, day, hour, minute, second, millisecond,
- TimePeriod – defining the period between two time instants and
- TimeDuration – defining time duration in terms of days, hours, minutes, seconds, and/or milliseconds.

The ontology *time* also defines the main time operations, i.e., it defines the operations to add/subtract time duration to/from a specific time instant, to check whether a time instant is within a defined time period, to check if the duration is longer/shorter than a specific duration, etc.

The location types and predicates are defined within the CoRaL in-built ontology *geo*. This ontology defines the location types *Location* referring to a location point (*latitude, longitude, altitude*), *GeographicArea*, referring to a geographic area (defining the border points of a polygon) and *Ellipse* referring to an ellipse area (specified by the centre point and the diameters). Moreover, this ontology defines the functions *locationInEllipse*(*Location,*

Ellipse) and *locatedIn*(*Location*, *GeographicArea*) that can check if the specified location is within the borders of an ellipse or a geographic area, respectively.

The ontology *basic_types* defines the types of *Frequency*, *Bandwidth, Power* encompassing the remaining DSA frequency and power related types. Furthermore, the ontology *powermask* provides options to define a specific power mask that can be used to limit the power emissions within the specified borders.

Having defined all the required frequency, power, time and location related types, CoRaL is generally a DSA policy language. In order to encompass the remaining CR parameters and provide the policy control of protocol stack parameters, one can introduce ontologies defining the physical, data link, network, transport, as well as application layer parameters. For instance, an ontology accounting for the PHY and MAC layers of the CR can specify the types of modulation and coding scheme, the type of the medium access control, the frame duration, Automatic Repeat reQuest (ARQ) procedures etc. The network and transport layer parameter types can be introduced in an additional ontology, which will cover parameters such as network and transport protocol version and types, routing types, as well as different data reliability and congestion control related parameters. The application layer ontology should support the definitions of the application types, priorities as well as QoS related parameters of application bitrate, delay, jitter etc.

For the purposes of the ARAGORN policy architecture and prototype [1], [2], the CoRaL policy language, besides the original CoRaL frequency, power, location and time-related ontologies elaborated before, was extended to support the specifications of the application types and priorities and radio access technology specification. Moreover, the definition of Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) MAC type was also introduced accounting for the backoff slots durations and frame size.

2.2.4 The IEEE P1900.5 Standard – Policy language requirements

The IEEE P1900.5 Standard has been officially released in mid-January 2012 [30]. The main goal of this Standard is to provide the policy language requirements and system architectures for DSA systems. In this section, the requirements and guidelines for policy languages defined in the IEEE P1900.5 Standard are presented, while detailed information about the system architecture is presented in Section 3.2.

2.2.4.1 Language expressiveness

In the Standard, general expressiveness requirements of a PL have been defined. These are summarized as follows:

- The PL should be declarative (not imperative) and should contain clear, unambiguous syntax and semantics.
- The PL should have the capability for annotations (also in natural language).
- The PL syntax should be understandable by machine and easily by human.
- It should support both permissive and restrictive policies.
- Inheritance and extension of policies should be able to be expressed in the standardized PL.

- The PL should be capable to specify dynamics (i.e., behaviour) of DSA system components.
- It should be possible to define new functions' definitions in terms of other known functions; also the PL should allow inferring relationships between functions.
- The PL should have the following elements used for general expressivity: classes, individuals, binary relations, composition of relations, functions, rules, state of the system, behavioural descriptions, and finally temporal aspects of the system.
- Two types of negation should be included, i.e., logic negation ("facts that have not explicitly been asserted to be true are not presumed to be false, they are simply unknown" [30]) and negation as failure (i.e., "statements that have not been asserted to be true and cannot be proved to be true are regarded as false" [30]).
- The PL should support variety of policies; the list of minimum set of 11 policies have been defined in the IEEE P1900.5 Standard.
- The PL should support expressing of meta-policies that may for example express relation between other policies; the PL should define at least one meta-policy that defines how to combine restrictive and permissive policies.
- The standardized PL should be able to express relation between policies and their compositions.
- Nested policies as well as policy templates should also be able to be defined.

2.2.4.2 Ontology requirements

In the Standard, some requirements of the PL ontology are provided. The ontology is defined in this Standard as the definitions that associate names of entities and concepts in a problem domain (e.g., objects, classes, etc.) with text describing what the names mean, and axioms (expressed in formal language) that constrain the interpretation of these entities and concepts [30]. In that light, the PL ontology shall be capable to express concepts required to encode regulatory, coexistence and system policies. It is assumed that the PL shall provide the ontological concept to describe information from at least three domains, i.e., the capabilities of the radio, the current environment of the radio and the characteristics of the requested transmission. Finally, the PL shall provide appropriate syntax and semantics for the representation of the selected specific concepts. The minimum set of concepts that have to be defined is also provided in the IEEE P1900.5 Standard.

2.2.4.3 Allowed expressions and supported data types

The PL compliant with the IEEE P1900.5 Standard shall provide appropriate operators to form various expressions. These operators can be classified as follows:

- Class of logical operators (at least and, or not, exists operators have to be defined).
- Class of mathematical operators (e.g., addition, subtraction, unary negation, multiplication, division, modulus, etc.).
- Class of relational operators (these allow for data comparison, e.g., less than, greater than, equal to, starts with, overlaps, etc.).
- Class of spatial operators (e.g., inside, distance).
- Set of operators – union, intersection, null.

- And finally invocation of procedural attachments.

The list of supported data types is also provided. These include:

- Scalars of various types (numbers – integers and floats, dates/time, Boolean, characters and text).
- Arrays and associate arrays.
- Enumerations.
- Terrain elevation Data.
- Spatial Objects (e.g., points, lines, shapes, such as polygons, polylines).

2.2.5 The ACROPOLIS approach

One of the major challenges of the ACROPOLIS WP 12 is to define the general characteristics and features of the language that will be used to express the policies that will be later applied on the developed decision making framework.

Based on the review of the state-of-the art in the area of policy languages proposed for CR systems, it can be stated that although various guidelines exist, such as XGPL or the recently released IEEE P1900.5 Standard, only one formal cognitive policy language that could be considered for implementation currently exists, i.e., CoRaL. Thus, CoRaL is the most promising candidate language to be used in the current and planned research within the ACROPOLIS project.

Taking into account the recent guidelines provided by IEEE, as well as the overall structure of the project and the direction of the on-going joint research activities, the focus will be on complying with the requirements introduced by the recent IEEE P1900.5 Standard. Let us stress, however, that the implementation of a specific, already defined policy language such as CoRaL, or the complete compliance with the IEEE guidelines will be avoided. The reason for this is that in such an approach, particular partners involved in standardization activities are able to provide new contributions, not standardized at the moment.

3. Policy framework standards

Following the description of the most notable existing policy frameworks and languages for CR systems, this section introduces the recent standardization efforts related to policy based decision making. To this end, this section provides an overview of the IEEE P1900.4 [12] and IEEE P1900.5 [30] Standards, introduced by the IEEE Dynamic Spectrum Access Networks Standards Committee (DySPAN-SC) [36].

3.1 The IEEE P1900.4 Standard

The IEEE P1900.4 Standard [12] mainly targets to architectural aspects of heterogeneous wireless networks. However, it has also incorporated policy aspects and this is the reason that is memorized hereafter.

3.1.1 System View of P1900.4

The high-level system architecture of the P1900.4 Standard is depicted in Figure 3-1. As can be seen in the figure, policies are exchanged between two entities, namely the NRM and the TRM. In particular, NRM provides TRM with the resource selection policies that the terminals have to be compliant with through a logical communication channel. The latter is known as radio enabler and can be mapped on one or more RANs which can be also used for data transmission. An example of such a logical communication channel can be described by the concept of CPC [37].

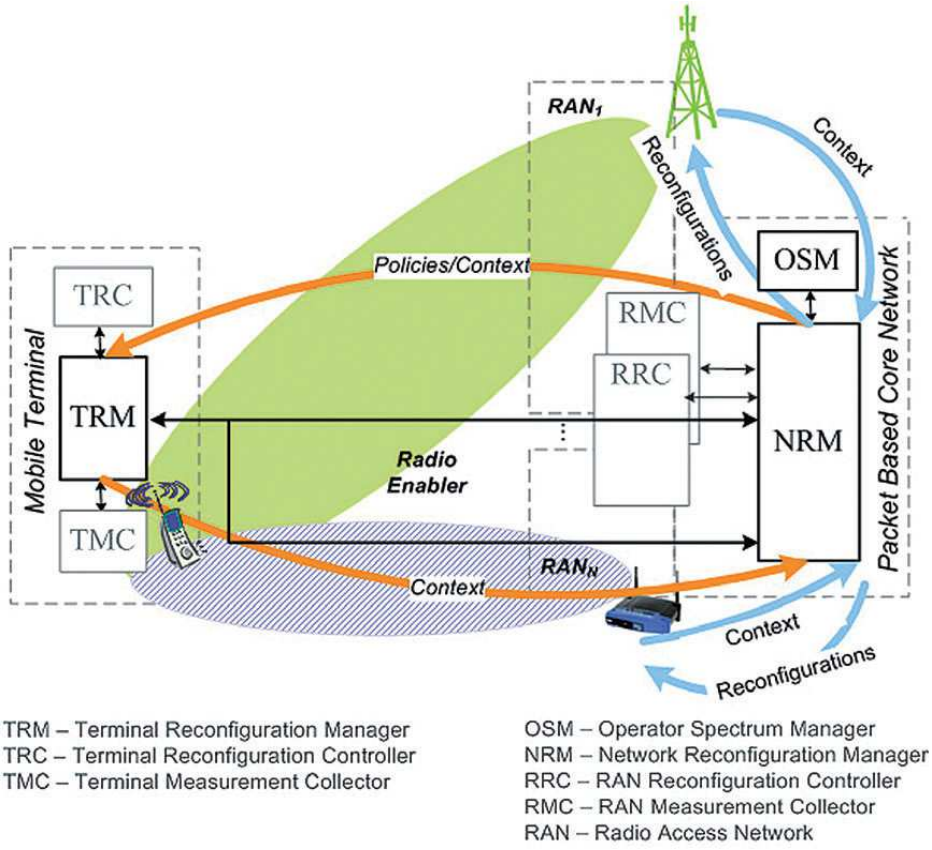

Figure 3-1: High-level system architecture of P1900.4 [18]

3.1.2 Policies and decision making in P1900.4

Moving a little deeper in the mechanisms of the NRM, policy efficiency and policy derivation functions are met. The first function is responsible for the assessment of the efficiency of the policies which is evaluated according to the collected information and based on various criteria. Some examples of the criteria may be the network quality or the user satisfaction. Accordingly, the respective metrics may involve the load, the throughput, the blocking and the delays. Eventually, the estimation of the efficiency of the policy is forwarded to the policy derivation function.

The policy derivation function decides on the necessity of deriving new policies. If the policies are found to be ineffective, the function derives them and sends them to a repository which, eventually, holds all the necessary information that can guide a mobile terminal to select the optimal RAN with respect to its demands. This repository is divided in two parts, the first of which is dedicated to policies, and thus called Policy Repository. The Policy Repository is also divided into two more parts, i.e., the basic-Policy Repository (b-PR) and the extended Policy Repository (e-PR) which hold the most important and the less important policies, respectively. Figure 3-2 gives an insight of the internal functions of a NRM and thus its functional architecture.

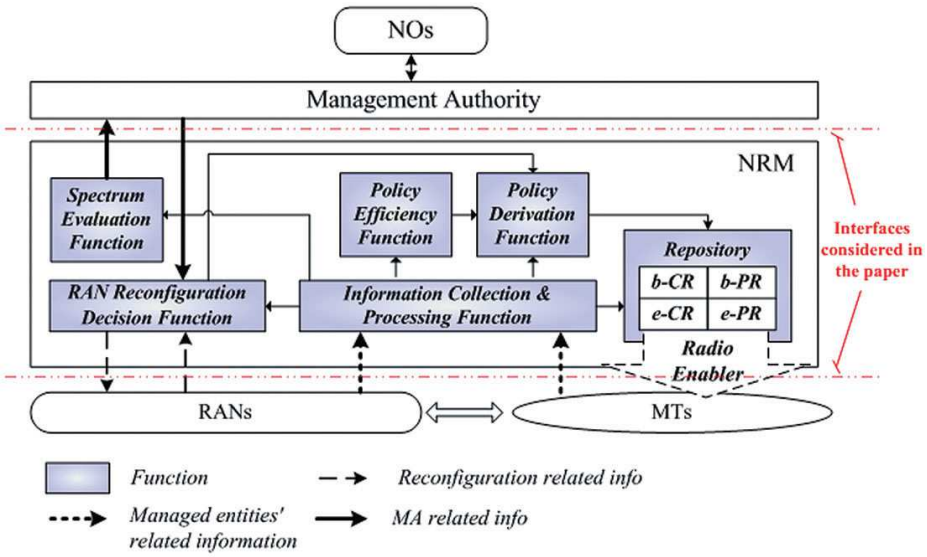

Figure 3-2: Functional architecture for NRM [18]

3.1.3 Information flow of policies in the context of P1900.4

As already stated above, policies are provided by the NRM to the TRM after being assessed by the policy efficiency function of NRM, derived by the policy derivation function of NRM and stored in the Policy part of the repository. Moreover, they target at assisting the terminal to reach their best connectivity.

To begin with, each policy is valid for a specific time period, is identified by a unique identification number (*pid*) and has a "grade of obligation" (*GoO*) which reflects the degree of importance of the policy for the behaviour of the corresponding terminal(s). In particular, the meaning of the values of a *GoO* for the targeted terminal(s) is the following:

- *GoO* = 1 means that the respective terminal(s) have to be compliant to this policy; usually applies in policies stored in the b-PR part of the repository.
- *GoO* = 2 names the policy as a policy that needs to be taken under consideration by the terminal(s).
- *GoO* = 3 gives to the policy an advisory character, i.e., it lets the terminal(s) decide whether they will take it into account or not depending on their own strategies.

Table 3-1 and Table 3-2 provide examples of structures of policies of both b- and e-PR. More precisely, Table 3-1 gives a general example of policy of b-PR with *pid x* and *GoO* equal to *y*. The terminals that are affected by this policy are those in ms_k with priority *pr* in time period (t_{i-1}, t_i), $i \in N$, which are guided to select ran_j as first choice for operating RAN, ran_m as second choice, etc. In critical situations, access of all terminals to any RAN is prohibited. Accordingly, Table 3-2 presents two examples of policies of e-PR where the identification number is *pid x* and the *GoO* equals to *y*. This policy suggests to terminals which are in ms_k with priority *pr* in time period *(ti-1, ti), i∈N* and are interested for service *s* to use frequency SU_f, which is available in this area for secondary usage, with power transmission $Pt \leq P_{Suf,max}$.

Structure of policies of b-PR	
General Policy	**Policy for critical situations**
pid x with GoO==y IF {ms_k AND [priority==pr] AND [TimePeriod==(t_{i-1}, t_i)]} THEN {1=ran_j, 2=ran_m, 3=ran_n ….}	pid x with GoO==y IF {ms_k AND [priority==any] AND [TimePeriod==(t_{i-1}, t_i)]} THEN No Access

Table 3-1: Examples of structure of policies of b-PR [18]

Structure of policies of e-PR	
General Policy	**Policy for spectrum secondary usage**
pid x with GoO==y IF {ms_k AND [priority==pr] AND [TimePeriod==(t_{i-1}, t_i)] AND [Service==s]} THEN {1=ran_j, 2=ran_m, 3=ran_n ….}	pid x with GoO==y IF {ms_k AND [priority==pr] AND [TimePeriod==(t_{i-1}, t_i)] AND [Service==s]} THEN SU_f with $P_{Suf,\ max}$

Table 3-2: Examples of structure of policies of e-PR [18]

3.2 The IEEE P1900.5 Standard

The IEEE P1900.5 Standard [30] addresses the policy language requirements and system architectures for DSA systems. In this section, the general architecture requirements for the policy-based control of DSA radio systems will be provided, followed by the description of the architectural components and interfaces. Also, the use case provided in this Standard will be briefly presented.

3.2.1 Target

In the IEEE P1900.5 Standard, the functional architecture of the policy-based DSA radio system (PBDRS) is considered. In that sense, the architecture is defined in the form of combinations of components (i.e., specific elements with interfaces and functionality) and functions (i.e., expected capability). This document provides a high-level description of the components and interfaces of the DSA policy framework and intentionally makes no assumptions regarding any physical implementations. By providing such a generic

description of the policy architecture, this Standard is mainly intended to minimize the vendor dependence, to define the main components and interfaces of this architecture, and finally, to identify and separate the functionalities of particular components.

3.2.2 General requirements for policy based DSA system architecture

The following general requirements for the PBDRS architecture have been defined:

- System strategy reasoning capability (SSRC) – the overall architecture should allow for system strategy reasoning;
- Presence of a Policy Conformance Reasoner (PRC) – within the architecture, a dedicated component for determining the compliance of the radio transmission comments with the existing (active) policies should be defined;
- Efficient ways of device accreditation – it is assumed that any PBDRS device shall be authorized and accredited of its policy conformance;

Also, some key-aspects of the requirements put on policy management have been identified, these are illustrated in graphical form in Figure 3-3.

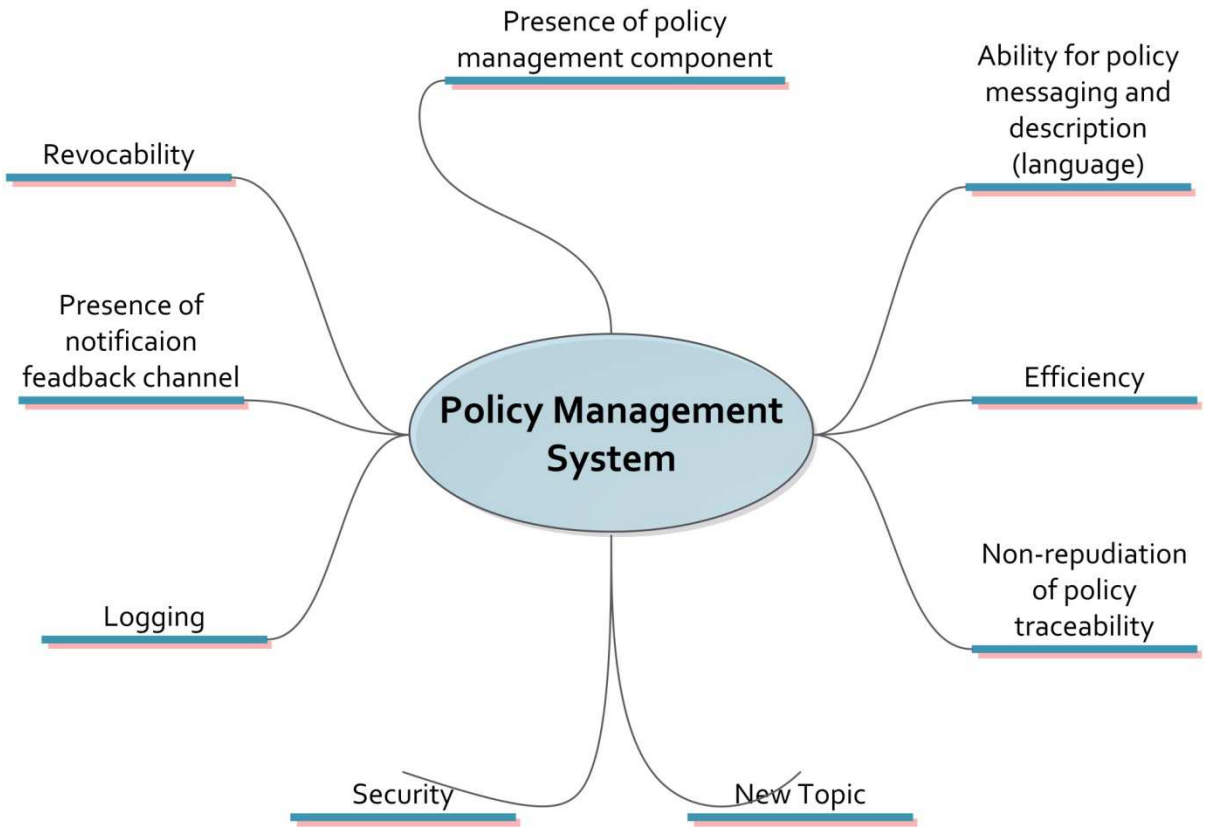

Figure 3-3: Requirements put on the IEEE P1900.5 policy management system

3.2.3 Description of main architecture components and interfaces

The schematic representation of the components and interfaces of the IEEE P1900.5 DSA policy architecture is presented in Figure 3-4. Let us highlight that the design of the components (blocks) and interfaces (connectors) are intentionally left for system designers for final decisions. These elements are also the focus of the Standard.

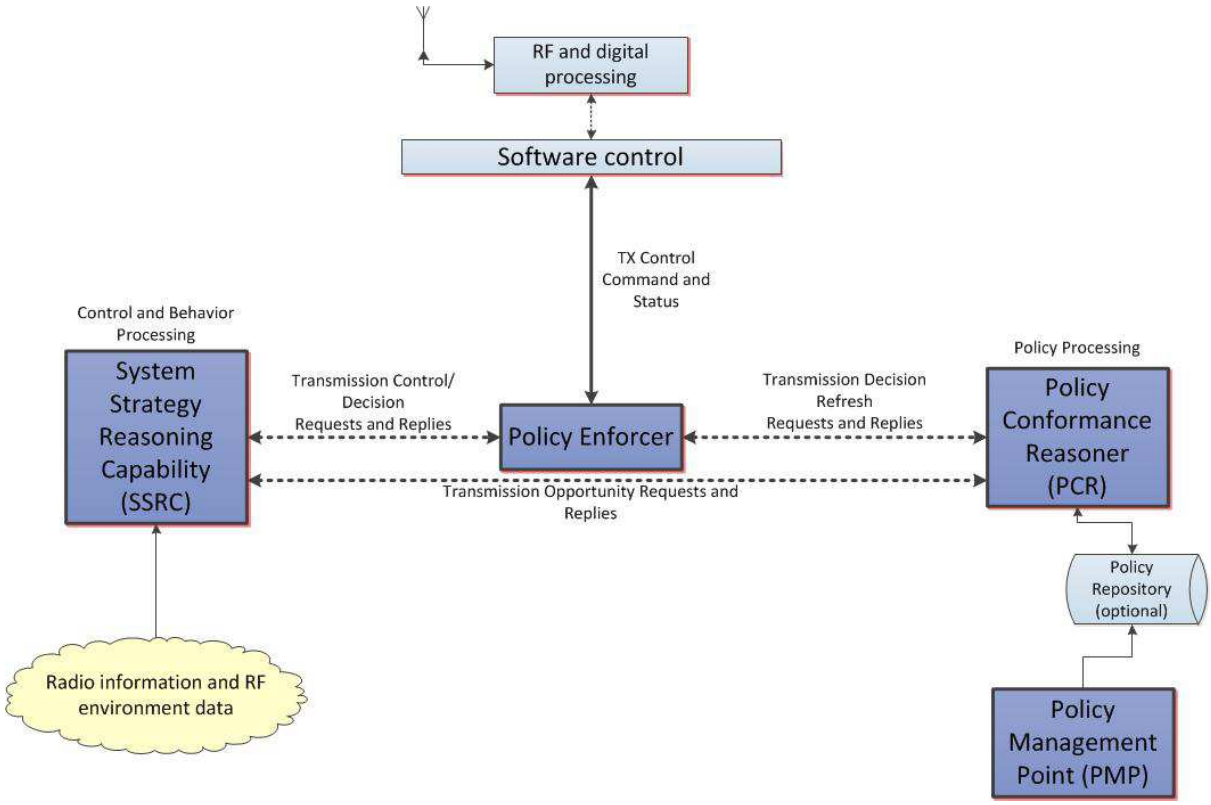

Figure 3-4: Generic architecture of the IEEE P1900.5 DSA policy framework

As can be stated, five separate components (i.e., one optional and four obligatory) have been identified. These are: SSRC, Policy Enforcer, PCR, Policy Management Point (PMP) and finally optional Policy Repository. Their main features, input-output data, as well as interfaces are summarized in Table 3-3.

Component:	**System Strategy Reasoning Capability (SSRC)**
Functionality	This component allows for system strategy reasoning. It is responsible for the system (re-)configuration. The SSRC shall have the possibility to learn from previous responses from other components. The main features of the SSRC are: • To be aware of the actual state of the radio and RF environment and use that information to formulate transmission strategies. • To submit transmission decision requests to the Policy Enforcer and react on its answers. • To revise its transmission strategies in response to replies from the PCR. • To send transmitter control requests to the *Software Control* component (via the Policy Enforcer).

Interfaces:	SSRC interfaces with the PCR and the Policy Repository components and receives Radio and RF information.
Inputs:	SSRC collects data about the Radio and RF environment (such as user identification, network information, local sensor information, databases, etc.); it also analyses responses from the PCR.
Outputs:	SSRC sends Partially and Fully specified transmission opportunity requests to the PCR as well as Transmission Control Requests.
Component:	**Policy Management Point (PMP)**
Functionality	The main goal of the PMP is to convert high-level policy information (understandable by human) to policies expressed in a standard policy language interpretable by various devices in the network.
Interfaces:	PMP interfaces with the PRC either directly or via the optional Policy Repository.
Inputs:	Policies defined in human language.
Outputs:	Policy (or set of policies) expressed in a standard policy language.
Component:	**Policy Conformance Reasoner (PCR)**
Functionality	PCR uses the logical system to make decisions regarding the conformance of the policies under investigation. It has to be able to: • Accept policy sets (sent from the PMP or the Policy Repository) and make all of the policy conformance decisions. • Verify if the transmission opportunity requests obtained from the SSRC comply with the active set of policies; if they do comply, then the PCR should accept this request. Transmission decision replies have to be provided to the Policy Enforcer and the SSRC. • Accept transmission decision refresh requests from the Policy Enforcer if they comply with policy rules from active policy set and provide answers to the Policy Enforcer.
Interfaces:	PCR communicates with the PMP, SSRC and the Policy Enforcer.
Inputs:	Inputs can be listed as: Transmission opportunity requests from SSRC; Transmission decision requests from the SSRC and sent via the Policy Enforcer, Transmission decision refresh requests from the Policy Repository; and finally policy management sets from the PMP.
Outputs:	Outputs can be listed as: Transmission opportunity replies to the SSRC and jointly to the SSRC and the Policy Enforcer, transmission decision refresh replies to the Policy Enforcer.
Component:	**Policy Enforcer**
Functionality	The Policy Enforcer shall be the point where the policy decisions are actually enforced. Thus, it shall examine the commands sent by the SSRC and will cause policy compliant commands to be delivered to the *Software Control* block. It shall send error messages if it cannot accept transmission control commands. Finally, the Policy Enforcer can ask the PCR to refresh its decision if the validity period has expired.
Interfaces:	The Policy Enforcer communicates with the PCR, the *Software Control* component and the SSRC.
Inputs:	The Policy Enforcer gets information from the PCR regarding the allowed transmission parameters. This information is provided in response to the SSRC transmission opportunity request or to the Policy Enforcer transmission decision refresh request.
Outputs:	The Policy Enforcer sends Radio Control commands to *Software Control* component,

	Transmission decision refresh requests to the PCR, and transmission control/decision replies to the SSRC.
Component:	**Policy Repository – optional**
Functionality	The policy repository acts as the logical container for various policies, it contains all policies required to verify the policy compliance
Interfaces:	Policy Repository takes information from PMP and replies to PCR.
Inputs:	The policy repository takes policy definitions from Policy Management Point
Outputs:	The Policy Enforcer outputs the policy which are compatible with the standard policy language

Table 3-3: Description of main components of the IEEE P1900.5 DSA architecture

4. Generic policy framework

The main disadvantage of the policy framework standards (IEEE P1900.4 and P1900.5) elaborated in the previous section lies in their architectural complexity and scenario (use-case) dependency. The main focus of this section is to elaborate and define a simpler and more synergic policy framework capable of covering all possible use-cases and providing a more unified and "easy to implement" system architecture. Moreover, the ACROPOLIS notion of the proposed policy framework classifies the CR devices in different classes based on their level of policy functionalities and argues on how policies can be deployed in a CR system while focusing on the decision making process and the decision making architectures. The generic policy framework can be consisted from four main components [38]:

- Policy Server (PS),
- Policy Manager (PM),
- Policy Engine (PE) and
- Policy Enforcing Point (PEP)

The **PS** represents the central storage device of the policies and the policy users. Its main task is the policy storage and distribution as well as user registration and user-policies associations. The **PM** is the central policy component in charge of the policy management. The main task of the PM is reading, writing, modifying and deleting the policies from the PS. The PM can either accept policies coming from a Policy Administrator (PA) or derive the policies from the gathered Radio Environment (RE) data that represents the behaviour of the CR system. The **PE** is the ***decision making enabler*** point in the policy framework and can be located on the user/terminal or network side. The PE consists of two main building components, i.e., a Policy Reasoner, responsible for reasoning of the active policies associated with a given user, and a Policy Database (DB), responsible for storing the operator/regulator and local user policies. The PE is responsible for presenting the reasoned results to the PEP. The **PEP** is the component of the architecture that is responsible for the execution of the policy rules by policy enforcement, context information gathering, optimization, learning, etc. The behaviour of the PEP can be monitored by the PM, from the gathered RE data, and can be used for derivation of new policies.

As illustrated in Figure 4-1, the elaborated policy framework yields interfaces that provide the necessary communication between all architectural components. The **PM/RE/PA** interface allows the PM to gather the needed information about the behaviour of the CRs thus deriving new policies that will enable more optimal operation of the cognitive system. The interface also allows communication between the PM and the PA. In this case, the policies are directly fed to the PM from the PA. The **PM/PS** interface defines the communication between the PS and the PM. It allows the PM to extract, store, modify or delete policies from the PS. The **PS/PE** interface is responsible for the proper distribution of the stored policies between the PS and the PE of the CR devices. This interface is additionally used for user registration, policy and notification exchanges. The **PE/PEP** interface supports the local communication between the PE and the PEP. The messages exchanged via this interface are policy queries and replies as well as policy change and emergency notifications. Additionally, this interface enables the policy based decision making process.

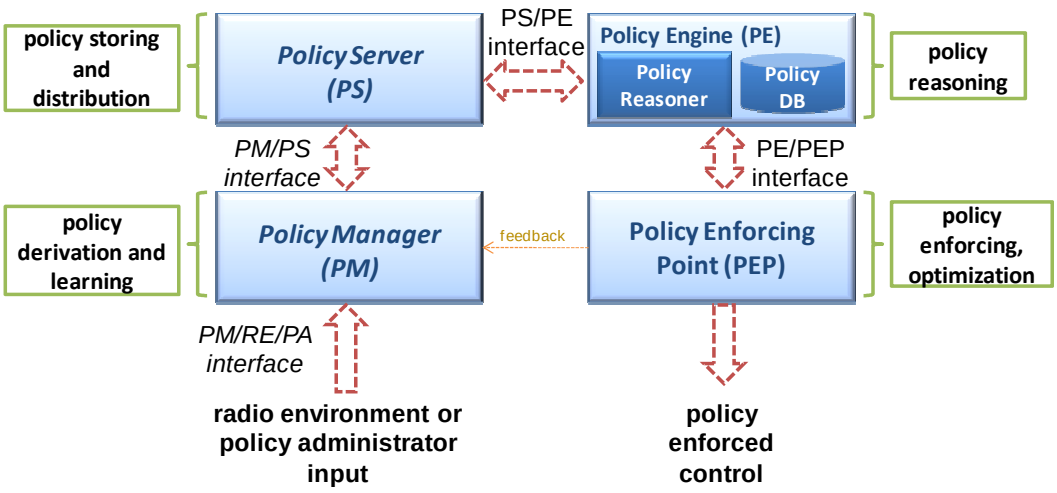

Figure 4-1: Generic Policy Framework [39]

Based on the level of policy functionalities, the terminals can be classified in two different classes:

- Policy controlled Cognitive Devices (PCDs) and
- Ordinary Cognitive Devices (OCDs).

The former ones incorporate both the PE and PEP components and are capable of standalone reasoning and enforcement. The devices of this type require higher hardware performance characteristics in terms of processing power, memory and battery capacity. The latter ones lack the PE component, thus require lower hardware performance characteristics. In this case, a remote reasoning option is feasible where the reasoning potential of a remote PCD node is utilized, see Figure 4-2.

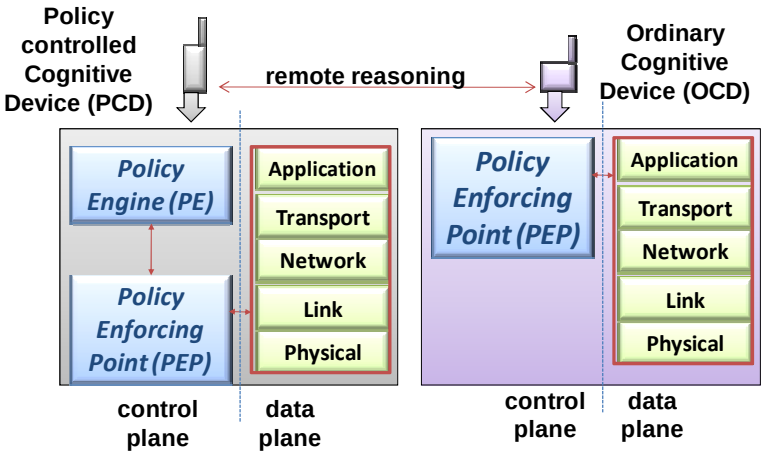

Figure 4-2: PCD and OCD terminals [39]

4.1 Policy based decision making

There are two main processes associated with the policy based decision making, i.e., the *policy reasoning* and the *policy enforcing*. The policy reasoning process refers to the extraction of an available and policy approved solution space based on the active policies and the present environmental conditions and preferences. The process of policy enforcing comprises the decision making tasks performed by the cognitive nodes in order to enforce the policy constraints into practice after selecting the best possible solution.

The basic concept of policy based decision making consists of two key components, i.e., the PEP and the Policy Decision Point (PDP), performing the above mentioned key policy related processes. Figure 4-3 illustrates this basic policy reasoning and decision making architecture for CRs including these two components.

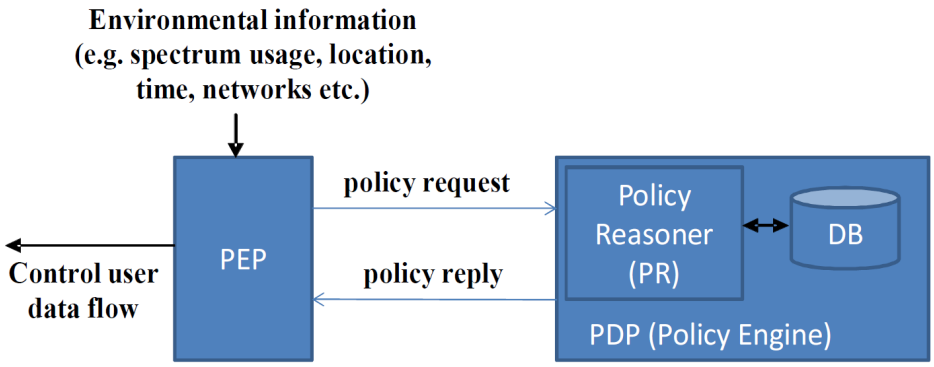

Figure 4-3: Basic policy reasoning and decision making architecture for CRs [40]

The PEP performs two important tasks, the first referring to the context information gathering and the second to the enforcement of policies. Namely, based on the environmental information, i.e., spectrum usage, location, available channels and networks and incorporated system strategies, the PEP creates and sends transmission queries to the PDP. Based on the policy replies received from the opposite direction, the PEP controls the radio transmissions, i.e., the user data flow. The PEP is placed inside every wireless node in a policy enforced CR system. One typical example of PEP is a *Radio Resource Management (RRM)* entity that optimizes the CR parameters in order to achieve better network performance.

The PDP is the second fundamental policy component in charge of checking the policy conformance of the requested transmissions by the PEP. The PDP can be also referred as a *Policy Engine (PE)*. It accepts the transmission queries, checks if the rules imposed by the active policies are satisfied, and returns policy replies to the PEP. Here, the actual reasoning is performed by a *PR* and all the relevant policies are kept and updated in a *Policy Database*. The PDP may not be present at all wireless nodes in a policy enforced CR system allowing for remote policy reasoning and lightweight implementations at certain nodes.

From the abovementioned points it is evident that the reasoning and enforcing process are tightly coupled and together they characterize the policy controlled decision making. Figure 4-4 depicts a policy based decision making architecture, where the RRM takes the place of the PEP. The RRM and PE utilize the PE/PEP interface for communication. This interface is used by both entities to exchange information about possible actions and constraints

defined in the user's policies. For example, based on its optimization problem, the RRM queries the PE about its intended actions. The PE responds with an answer whether the planned actions by the RRM are conforming to the policies or not. In this manner, the PE defines the boundaries of the solution space in which the RRM performs its decision making process and optimization.

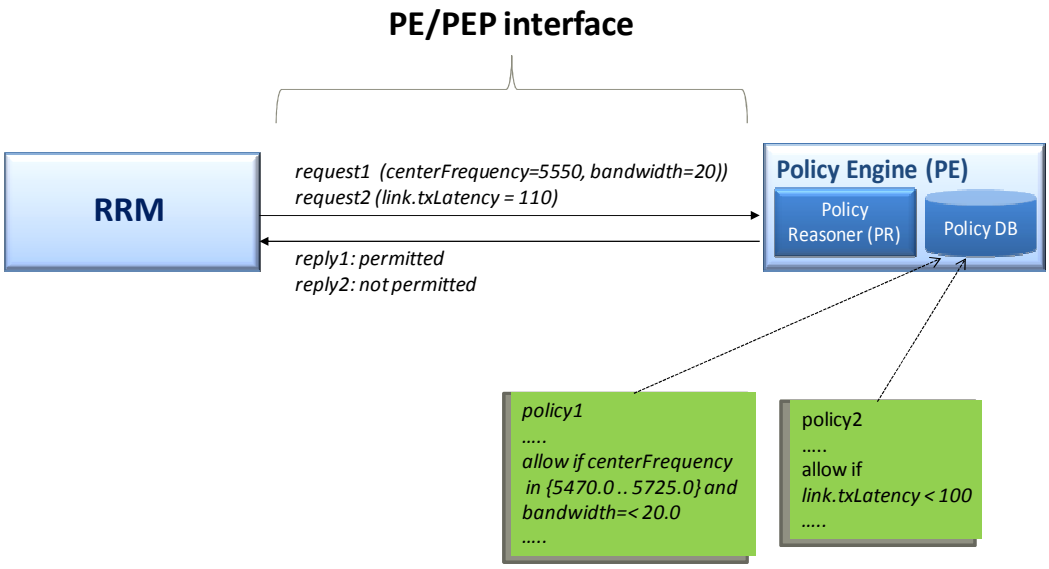

Figure 4-4: Generic policy based decision making [2]

Additionally, Figure 4-5 illustrates the Message Sequence Chart (MSC) of a policy based decision making scenario. After a CR node (user) has registered to the network, the PE module gathers and stores its policies. When the user's RRM needs to make a given optimization, first it queries the PE about the possible limitations. Based on the reply obtained from the PE, the RRM calculates the available solution space and makes the most optimal decision.

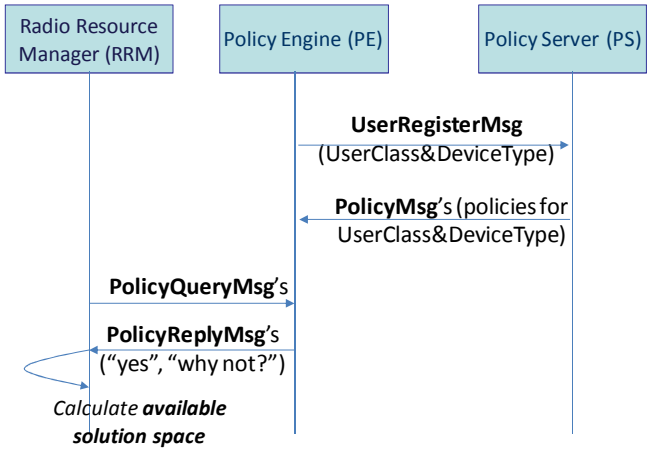

Figure 4-5: Policy based decision making MSC [1]

4.2 Policy based decision making architectures

Based on the location of the PE entity (i.e., the reasoning process within) there are three types of policy reasoning and decision making architectures:

- centralized,
- distributed and
- hybrid architectures.

Figure 4-6 depicts the centralized architecture. In this case, there is a centralized reasoning being performed by the serving Point of Attachment (PoA) for all active users. The centralized approach is more suitable for scenarios where the CR devices lack high-end hardware capabilities (i.e., processing power, memory, battery capacity, etc.) as in the case of the OCD type of devices. The negative side of this approach is the control plane latency due to the remote reasoning. Additionally, the latency will tend to increase as the number of active CR devices increases, resulting in a highly congested network or overloaded PoA that can degrade the performance of the CR system.

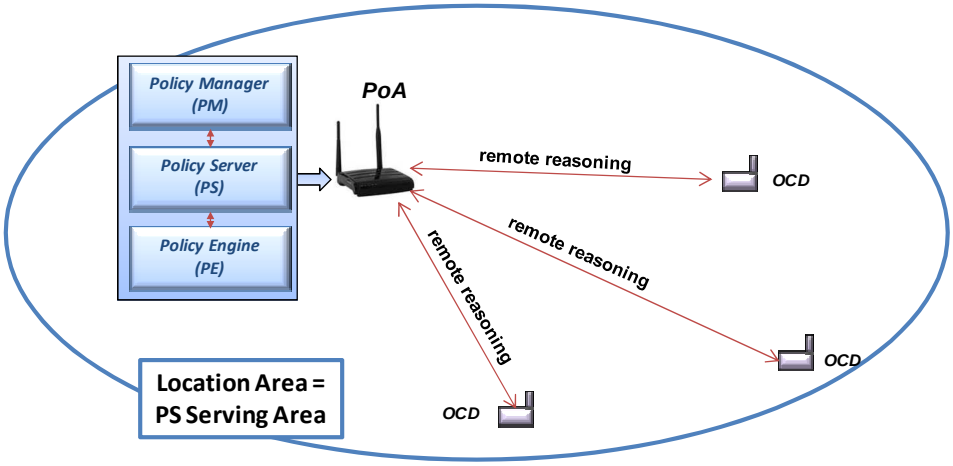

Figure 4-6: Centralized architecture

On the other hand, the distributed architecture is more suitable for scenarios where the PCD devices are introduced, see Figure 4-7. The distributed approach mitigates the latency effect, due to the local reasoning, at a cost of increased device complexity.

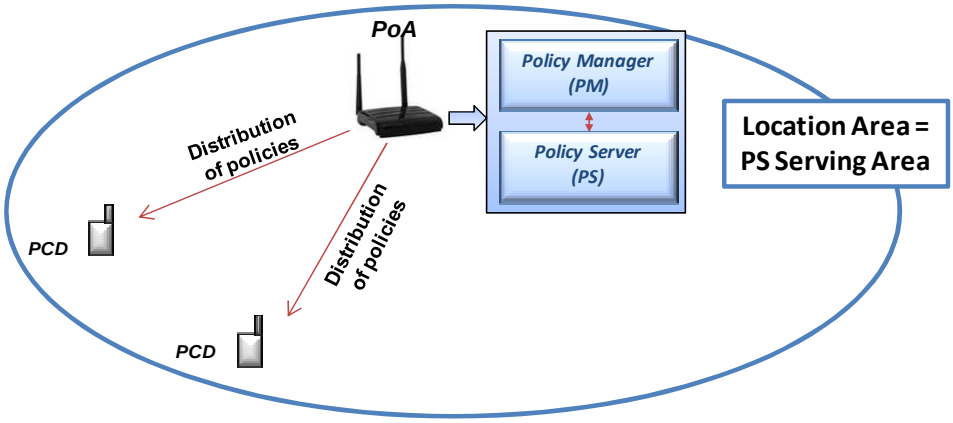

Figure 4-7: Distributed architecture

Figure 4-8 shows a possible hybrid decision architecture. In this case, the OCD devices can utilize the reasoning capabilities, in an ad-hoc fashion, of a nearby PCD, thus avoiding to overload the serving PoA and congest the network.

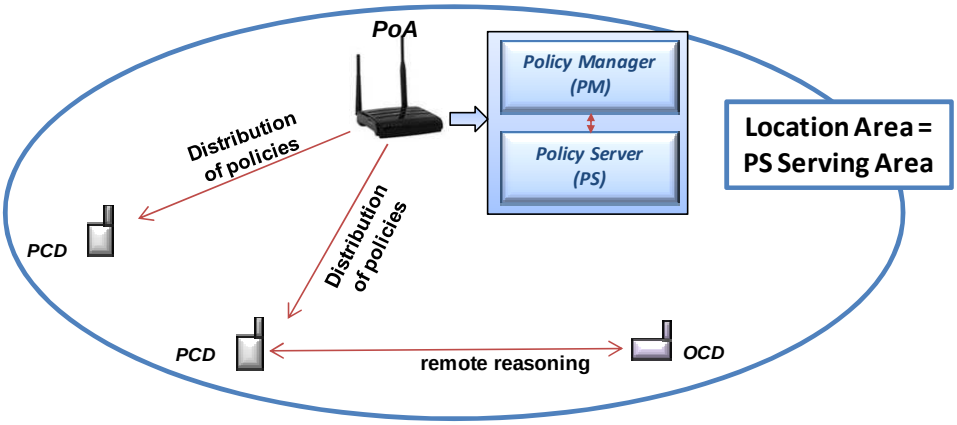

Figure 4-8: Hybrid architecture

4.3 Conclusion

The policy framework elaborated in this section differs from the related policy standards explained in Section 3 in the degrees of freedom it supports. Namely, the flexibility of the presented approach lies in the support of different types of cognitive terminals in the network, policy-enabled or non-policy-enabled ones, as well as on the architectural simplicity, modularity and generality. These options provide the possibilities to easily apply and adapt the presented framework to a various set of cognitive use cases to control the decision making process in a centralized, distributed or hybrid fashion.

5. Hardware platform requirements and constraints

This section describes the general requirements and constraints of a CR instantiation platform and elaborates on the applicability of the USRP2 platform on policy based CR scenarios, discussing on its features, hardware constraints, and the policy implementation issues. It aims at providing a link between the policy frameworks discussed in the previous sections and the operation characteristics of the actual experimental instantiation platforms by highlighting the requirements that have to be taken into account in order to implement a policy framework within a hardware platform.

5.1 Platform requirements

5.1.1 Required functional components, features and characteristics

One of the challenges in building an instantiation platform for experiments is defining its specifications, which must not only accommodate all current ideas but also ideas which have not yet been conceived [41]. The platform can be future-proof by adopting a modular architecture, with high performance components in each module. According to [42], central to all cognitive wireless protocol innovations are three functional components:

- **Sensing:** At the heart of cognitive wireless is the ability of nodes to sense their surrounding environment. Nodes should be able to use the observed spectral usage at multiple time-scales to adjust their operation accordingly. Furthermore, the nodes could report their spectral measurements to a server that may execute functionalities over a wider spatial scale.
- **Adaptation:** Based on the sensed environment, nodes should be able to adapt any part of their communication stack. For example, they could adapt their physical layer or medium access layer to be more conservative if they sense a primary network with which they must not interfere.
- **Co-ordination:** Since nodes in a wireless network experience different environments (due to propagation effects of the wireless medium), their local knowledge is different. As a result, they have a different view of the network and spectrum utilization. In order to achieve their system-wide utilization maximization, the cognitive nodes need to agree on their collective actions.

In addition to enabling the above basic functionalities for cognitive protocol innovations, an experimentation platform –which should be completely programmable and easily deployable- has to enable three additional features:

- **Isolated and Integrated Experimentation:** The platform should enable isolated experiments, which are essential for systematically testing sub-components of a larger system. Yet, at the same time, the platform design flow should enable seamless integration of sub-components into fully functional networks of nodes.
- **Programming and Control of Deployed Networks:** To continue innovation in a deployed network of programmable nodes, it is crucial that the rapid reprogramming of each node is possible. Better still, the nodes should be controllable in real-time to allow the complete exploration of the design space, as this is defined by the implemented protocols.

- **Detailed Observability of the Operational Network:** Finally, each node should be able to report measurements from any layer to enable a deep inspection of any part of the network stack. Such a capability is essential to provide insights into the operation of clean-slate, cross-layer designs and crucial to optimize networks for achieving the highest performance.

In [42], existing experimental platforms and testbeds of CR ad hoc networks are classified and reviewed. Typical existing platforms are based on a split architecture, where a reconfigurable generic radio device is connected to a host, typically a desktop or a laptop computer. The key resource allocation and digital signal processing (DSP) operations are undertaken on the host, which is connected to the CR device through a high-speed connection, e.g., Universal Serial Bus (USB) or Gigabit Ethernet. The host contains implementation of the communication protocol stack, which handles data at the packet level, and of the PHY functionalities. The output of the processing on the host is a digital sample stream that is transferred to the CR device for further processing.

A typical Reconfigurable Board consists of a motherboard with plug-in transceivers that permit incoming RF signals to be digitized and generates outgoing RF signals from the digital sample stream sent by the host computer. The platform typically includes an embedded FPGA, allowing processing in hardware of the samples on the ingress and egress paths. Interchangeable daughterboards can cover different frequency ranges. Digital/Analog conversion is realized either on the FPGA, or on dedicated hardware. For example, the USRP2 mounts a specialized 400Msamples/s 14- bit DAC, and 100Msamples/s 16-bit ADC.

A soft-core processor may be implemented on the FPGA to handle internal board operations, e.g., controlling arbitration on the data bus. For example, in USRP2 a 32-bit soft-core processor (AeMB) is implemented on the FPGA and handles most of the internal USRP2 operation. Note that in the basic USRP2 configuration, the AeMB does not perform any DSP functions; these are done purely in software in the DSP receive and transmit pipelines on the host. Additionally, a CR platform will embed fast access memory (in the order of a few Mbytes), which can be implemented on the FPGA itself; and larger low-cost memory for long-term storage, for example through an external Secure Digital (SD) card.

The CR platforms and testbeds can be classified according to some specified characteristics [42], [43]:

- *Accessibility:* The overhead, both in terms of the material and manpower required for a thorough experimental evaluation, is often a detrimental factor in building testbeds. Several institutions provide external interfaces, where the individual devices of the testbed can be programmed by a user remotely, often over the Internet.
- *Device Hardware:* There are commercially available device types, as well as custom designs that may support (i) varying numbers of transceivers, (ii) processors clocked at different instructions/per second, (iii) frequency ranges and bandwidths that are supported by these transceivers, (iv) antenna types, among others. Ideally, the testbed should be highly re-configurable in terms of dynamic spectrum selection, and be able for wide-band sensing over large ranges.
- *Scale:* Depending upon the resource constraints, testbeds vary in the number of nodes, ranging from around a dozen to nearly 50 nodes [41]. Specifically, large testbeds allow for realistic testing of the effect of the interference caused by intra-

CR network transmissions on licensed user detection, and the performance of spectrum sharing in the detected vacant bands.

- *Heterogeneity:* Most of the existing testbeds are composed of homogenous devices, and each node has similar communication and processing capabilities. However, real-world networks are likely to be highly diversified in these respects. Hence, devising a compatibility plane is needed, where individual nodes offer additional and often distinct capability over a minimum acceptable feature set. Such testbeds may also be used to test the coexistence of CR networks managed by different entities in a common spectrum pool.
- *Protocol Support:* Given the high degree of reconfigurability in the CR devices, developing a user modifiable protocol stack that can run on the devices is a challenge. When research efforts are undertaken at each layer, basic implementations of the other protocols layers, not in the scope of the work, must be present. As an example, Transmission Control Protocol (TCP) implementations over CR may rely on spectrum sensing information from the physical layer. Moreover, the basic structure should be easily modifiable, as there has been an increasing trend towards developing cross-layer techniques that integrate and utilize the complete spectrum as well as network information.

5.1.2 Requirements of a cognitive specification language

In [44] the limitations of a selected platform (USRP with software support provided by GNU Radio) were highlighted and ideas of what would be required to make it more broadly useful for cognitive networks research were discussed. Four aspects are of particular importance were identified:

- Support for distributed signal detection and modulation classification algorithms to allow the assessment of the effectiveness and requirements of distributed methods in detecting the presence of incumbent users and utilizing white spaces;
- Support for a MAC protocol tailored to the requirements of dynamic spectrum access and in particular one that supports random access;
- Development and implementation of a Network Knowledge Representation Language (NKRL) to store and communicate information regarding network state to cognitive elements; and
- Development and implementation of a cognitive specification language (CSL) that describes the interface between policies and objectives and the cognitive engine.

The first two aspects address usability of an open CR platform in a wireless network, while the last two refer to the reusability of cognitive engines.

As mentioned in the last aspect, CSL describes the interface between policies and objectives and the cognitive engine/decision engine, where cognitive process/decision-making is carried out. Either the operation of the cognitive/decision engine and the execution engine is decoupled by constraints and requirements formulated by the decision engine, or the decision and execution tasks are jointly optimized. In the first case, the decisions can be made without considering a certain hardware platform or transmission technique of the cognitive system. The task of the execution engine is then to map the constraints to the specific platform and to the available transmission techniques. In the second case, a close

cooperation between both engines is required and both complexity as well as performance are increased.

The CSL is analogous in scope and intention to a QoS specification language. These languages are used to represent QoS requirements to the various mechanisms that the network offers to support them. There are already several different existing QoS specification paradigms and the concept of these languages – mapping requirements to underlying mechanisms – is the same here, except that the mechanisms are adaptive to the network capabilities as opposed to a fixed set of QoS capabilities. The following criteria represent design objectives for an effective CSL. A language that does not meet any or all of these requirements may still perform the role of a CSL, albeit less effectively [44].

- **Expressiveness:** A CSL must specify a wide variety of end-to-end goals. It should be capable of expressing constraints, goals, priorities and behaviours to the cognitive elements that make up the process. It should be able to express new goals without requiring a revision in the language.
- **Cognitive process independence:** The cognitive process architecture and functionality should not dictate the CSL. Instead, the CSL should abstract as much as possible the specifics of the cognitive process from the application, user, or resource. This allows a goal to be used in different cognitive processes with little modification and promotes re-usability.
- **Interface independence:** Whether the cognitive process is distributed or centralized in operation, autonomous or aggregated in architecture, it should be independent of the underlying interfaces. As mentioned in the previous point, this abstraction promotes reusability by allowing the re-use of goals over many different cognitive processes with little effort from the upper layers.
- **Extensibility:** The CSL should be extensible enough to adapt to new network elements, applications and goals, some of which may not even be conceived yet.

The effectiveness of an open CSL that accomplishes the abovementioned design requirements is determined by selecting several basic end-to-end objectives and cognitive processes (with differing interfaces and functionalities), and then having the CSL represent these objectives within the process.

5.1.3 Requirements regarding the policy framework

From the Policy Framework and Policy Enforcement Mechanisms point of view, it should be technically possible to use and implement one of the developed policy languages (such as CoRaL), and in general policy mechanisms. It means that the CR terminal (emulated in the existing hardware platform) should be able to adapt itself to the requirements provided by a policy reasoner and it has to be able to communicate and negotiate with the reasoner. This can be illustrated in the following example: The CR terminal asks the policy reasoner for the permission for transmission at the given frequency band and provides basic information about the sensed neighbouring terminals. The policy reasoner replies in the conditional manner asking for some additional measurements (e.g., to scan wider frequency bands, to provide more accurate sensed parameters, etc.) and the cognitive terminal should be able to provide the expected data since otherwise it will be not permitted to transmit. If the cognitive terminal is not able to provide the necessary data on its own it could communicate

with the neighbouring sensors in order to get the required information. Based on the above example the following conclusions can be drawn:

- It should be possible to implement the required software environment for the application of one of the CR languages; the hardware platform shall be able to understand, interpret and create the declarative instructions, typical for such class of languages and to enforce some requirements toward the storage elements, used processors, etc.
- Since the cognitive languages are based on ontologies, the hardware platform shall be able to store and understand the used ontologies, enable their updates and modifications and to ensure fast and reliable access to them.
- The platform shall be adaptive in such a way that will allow providing the parameters required by the policy reasoner; such a statement imposes high requirements on the quality and performance of the particular elements used in the cognitive front-end architecture, e.g., the antennas should be either wideband or parametrically steered within the wide range of frequencies, the (Inverse) Fast Fourier Transform, (I)FFT, block (if needed) should be able to change the FFT size etc.
- The assumption of the CR's ability for executing the policies provided by the reasoner means that most of its elements shall be tuneable (in terms of various parameters as well as in terms of the wide range of values of these parameters, e.g., the attenuation band of the transmit shaping filter, the sampling frequency of the analogue to digital and digital to analogue converters etc.).
- In case the platform itself is not able to provide the necessary information to the policy reasoner, it should communicate with the neighbouring sensing nodes (separate sensors, other nodes or users) to get the expected data. It means that the hardware platform should be able to implement and use protocols like IEEE 1900.6 [45] which defines the way of communication between the CR network elements in a hardware-agnostic manner.
- The assumption of the sensing capability of the cognitive platform means that it should be able to implement sophisticated sensing algorithms that are characterized by, e.g., high requirements regarding the sensitivity levels, low noise-figure, high analogue to digital resolution and processing speed etc.
- Beside the 1900.6 protocol, the platform should be able to make use of the geolocation databases and of the specific communication protocols required for its connection to these databases and to any external information system with valid context information;
- Since one of the key-factors is the accurate localization of the terminal itself as well as of the primary users, the hardware platform shall be able to utilize the Global Positioning System/Global Navigation Satellite System (GPS/GNSS) signals or to exploit specific RAN functionalities and signalling that will assist in location determination.

5.2 Instantiation on existing Cognitive Radio platforms

In order to provide a concrete case study of the general considerations introduced in the previous section, this section elaborates on the applicability of the USRP2 [6] platform in policy based cognitive scenarios, its advantages and weaknesses in terms of the different

policy functionalities, i.e., the policy derivation, reasoning, enforcing, etc. USRP [46] is one of the most popular and widely used SDR platforms. The hardware flexibility and the availability of various software packages (including open-source solutions) make the USRP platform feasible for the practical realization and evaluation of different CR mechanisms and protocols. USRP2 is the second version of the USRP platform, updated with more powerful hardware and features compared to its predecessor.

5.2.1 USRP2 Overview

The USRP2, as well as the other SDR solutions provided by Ettus Research, comprises two main hardware elements (Figure 5-1 [46]), i.e., a daughterboard, representing the RF part, and a motherboard, performing the conversions between the analog and digital domains, as well as the operations of digital down- and up-conversion. The whole software processing is performed on the host computer side.

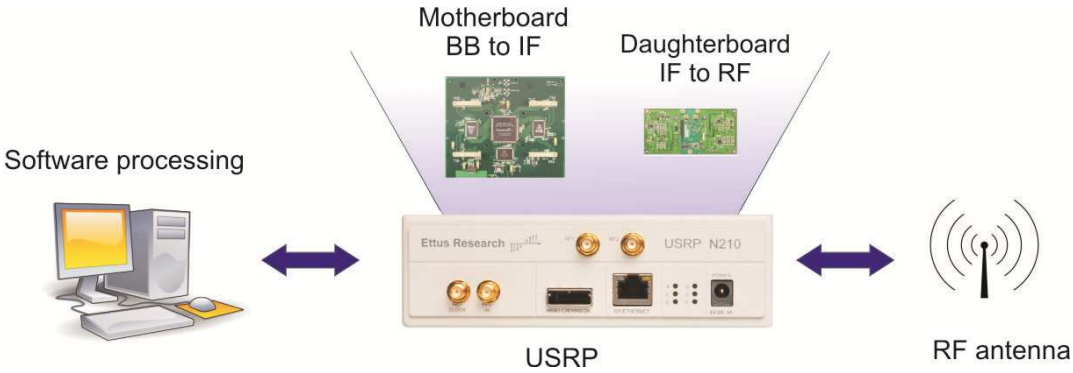

Figure 5-1: Generic diagram of the USRP-based wireless communication transceiver [46]

5.2.2 USRP2 Features

This subsection presents the USRP2 platform characteristics related to policy controlled cognitive networks. It discusses the sensing related features and the reconfigurable (policy-controlled) communication parameters of the USRP2 device.

5.2.2.1 USRP2 Sensing Features

The sensing capabilities of the USRP2 device can be utilized in the policy derivation and the policy reasoning tasks in the policy based cognitive network scenarios. The first task refers to the case when the sensing input from the USRP2 device (e.g., spectrum availabilities, statistics of the spectrum usage, etc.) is used for the derivation of spectrum policies. The second task refers to the case when a single USRP2 device, used as a cognitive device, exploits its sensing input to reason whether a specific band can be utilized or not.

The most prominent sensing features of the USRP2 devices are as follows.

Frequency resolution. The USRP2 device can perform the sensing with various RF bandwidth sizes ranging between 195 kHz and 25 MHz. The frequency resolution can be further increased if applying FFT.

Time resolution. The frequency switching (phase-locked loop - PLL) delay of the USRP2 device is below 200 μs and the samples acquiring delay is in order of tens of microseconds, which is sufficient for a large set of the cognitive scenarios. However, this is a major drawback in strict delay-sensitive scenarios.

Sensitivity and precision. The sensitivity (noise floor) of the USRP2 device is daughterboard specific, i.e., the RFX daughterboard has a noise floor of -164 dBm/Hz while the WBX daughterboard has a noise floor of -167 dBm/Hz. It should be noted that these numbers are specific for the highest receiver gain settings of the daughterboards. The data quantization is performed with 14 bits for each, I and Q samples, providing a sufficient resolution. However, one disadvantage of the USRP2 platform is the non-linear input-output characteristic, which can be a significant drawback in scenarios where the precise level of the primary signal is required. This issue can be overcome with calibration of the USRP2 level with a known signal source.

Detection methods. The basic sensing method implementation of the USRP2 device is the energy detection. Other detection methods, including feature detection (such as cyclostationarity detection) as well as different cooperative sensing methods are also feasible with this SDR platform.

5.2.2.2 USRP2 Communication Features

The USRP2 communication features are limited to the PHY layer in general. This is a result to the Ethernet connectivity of the USRP2 motherboard to the host Personal Computer (PC) side, as well as the inapplicability of custom made codes on the FPGA of the device. Namely, the delays between the USRP2 hardware and the output of the host PC application can be in order of hundreds of microseconds, which is inappropriate for MAC layer functions in general. The lack of available on-board memory (only 3% free) of the USRP2 motherboard, practically makes it impossible to implement code on-board and eliminate the high delays. Therefore, the USRP2 device communication capabilities are bounded only on PHY layer capabilities, i.e., the control of frequency, power, bandwidth, which is sufficient for most of the DSA scenarios.

The possible policy controllable communication parameters of the USRP2 device are:

- Frequency – depending on the used daughterboard, the frequency can be easily changed on-the-fly by the USRP2 device.
- Bandwidth – as mentioned, there is a flexibility in terms of the RF frequency band, it can be chosen in the range 195kHz – 25 MHz.
- Transmit power – the maximum transmit power is limited by the choice of the daughterboard. The power adaptation (transmit gain control) is in general not supported by the Ettus Research provided daughterboards, but can be achieved with digital scaling of the samples.
- Modulation and coding – the available modulation and coding schemes depend on the used software for the USRP2 platform [48]-[50]. However, there is a limitation on

the Orthogonal Frequency Division Multiplexing (OFDM) usage, due to its strict timing requirements and the abovementioned timing issues of the USRP2 platform.

- Multiple Input Multiple Output (MIMO) usage – the USRP2 makes it possible to use MIMO by interconnection of multiple devices. The transmit/receiving beamforming is inherently possible.

MAC implementations – as previously mentioned, due to the delay issues most of the time-sensitive MAC implementations are inapplicable to the USRP2. Still, one can use MAC schemes like CSMA, Time Division Multiple Access (TDMA), etc., taking into account the time limitations – using sufficiently long time slots.

5.2.3 USRP2 Constraints

Based on the previously elaborated features of the USRP2 platform, this part summarizes the constraints and the limitations of the platform.

In terms of the sensing, the USRP2 platform provides sufficient frequency resolution and sensitivity of the device. It also allows for the implementation of more complex detection algorithms. However, the main sensing limitations of the platform are the timing issues, i.e., the large delays between the USRP2 motherboard and the host PC resulting in delay of the sensing data. This, as well as the non-calibrated input-output characteristics, can be a significant drawback in certain policy based cognitive scenarios.

The timing issues of the USRP2 platform do not allow integration of time-sensitive MAC protocols on the platform making it possible only for PHY layer parameters to be controlled via policies. However, this is sufficient for a significant number of practical scenarios.

5.2.4 Implementation of policy features on USRP2 platform

Depending on the chosen software, the implementation of the policy functionalities on the USRP2 can be a challenging task. The different software packages available for the USRP2 device provide different set of signal processing blocks, and different possibilities for modifications and extensions. Table 5-1 presents the possible software solutions for USRP2 device with brief notes on the policy implementation possibilities.

Name / Feature	GNU radio [48]	LabView [49]	Matlab Simulink [50]
Processing blocks availability	Large set (modulations, coding schemes, filters, FFT etc.)	Large set (modulations, coding schemes, filters, FFT etc.)	Large set (modulations, coding schemes, filters, FFT etc.)
Open source (possibility of modification and extension)	Yes	No	No
Available implementation examples	Various	Few	Few
Complexity of policy implementation	Moderate	High	High

Table 5-1: Software packages for USRP2 and possibilities for policy implementation

6. Conclusion

This deliverable provided a summary of the scope, functions and basic mechanisms of the ACROPOLIS decision making framework.

First, an overview of the most prominent policy frameworks for CR networks, namely the ARAGORN, E2R, E3, ORACLE, and IEEE 802.22 WRAN frameworks, was provided, summarizing their main architectural components. The most well-known policy languages for CR systems, namely XGPL, ORACLE PL and CoRaL, were also reviewed and their basic structures as well as exemplary policy rules were provided. Moreover, the basic requirements for a policy description language, as described in the recently introduced IEEE P1900.5 Standard, were provided.

A brief overview of the IEEE P1900.4 Standard was provided, describing its system architecture, the policy and decision making process, and the information flow between the different system entities. Similarly, the general architecture requirements for the policy-based control of DSA radio systems, followed by the description of the architectural components and interfaces of the IEEE P1900.5 Standard was provided.

Then, the main components as well as the basic policy reasoning and decision making architecture of the ACROPOLIS notion of a generic policy framework was described. Such a policy framework differs from the related policy standards reviewed in the degrees of freedom it supports. More specifically, the flexibility of the presented approach lies in the support of different types of cognitive terminals in the network, policy-enabled or non-policy-enabled ones, as well as on the architectural simplicity, modularity and generality. These options provide the possibilities to easily apply and adapt the presented framework to a various set of cognitive use-cases to control the decision making process in a centralized, distributed or hybrid fashion.

Finally, the general requirements and constraints of a CR instantiation platform were provided and, in order to provide a concrete case study of these general considerations, the applicability of the USRP2 platform on policy based CR scenarios, its features, hardware constraints, and the policy implementation issues were discussed.

Glossary and Definitions

Acronym	Meaning
(I)FFT	(Inverse) Fast Fourier Transform
ADC	Analog-to-Digital Converter
AP	Access Point
ARQ	Automatic Repeat reQuest
BLM-REQ	Bulk Measurement Request
b-PR	Basic Policy Repository
BS	Base Station
CC	Control Channel
CLIPS	C Language Integrated Production System
CoRaL	Cognitive Radio Language
CPC	Cognitive Pilot Channel
CPE	Customer-Premises Equipment
CPS	Cognitive Pilot Server
CPU	Central Processing Unit
CR	Cognitive Radio
CRM	Cognitive Resource Manager
CSL	Cognitive Specification Language
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
CW	Context Watcher
DAC	Digital-to-Analog Converter
DARPA	Defence Advanced Research Project Agency
DB	Database
DBS	Database Service
DSA	Dynamic Spectrum Access
DSP	Digital Signal Processing
E2R	End-to-End Reconfigurability
E3	End-to-End Efficiency
EIRP	Equivalent Isotropic Radiated Power
e-PR	Extended Policy Repository
FP6	6^{th} Framework Program
FPGA	Field Programmable Gate Array
GNSS	Global Navigation Satellite System
GoO	Grade of Obligation
GPS	Global Positioning System
IEEE	Institute of Electrical and Electronics Engineers

IEEE DySPAN-SC	IEEE Dynamic Spectrum Access Networks Standards Committee
ISM	Industrial Scientific and Medical
MAC	Medium Access Control
MIMO	Multiple Input Multiple Output
MSC	Message Sequence Chart
MTA	Mobile Terminal Assignment
NKRL	Network Knowledge Representation Language
NRM	Network Reconfiguration Manager
OCD	Ordinary Cognitive Devices
OFDM	Orthogonal Frequency Division Multiplexing
OO	Object Oriented
OR	Opportunistic Radio
ORACLE	Opportunistic Radio Communications in unLicensed Environments
OWL	Web Ontology Language
PA	Policy Administrator
PBDRS	Policy Based DSA Radio System
PC	Personal Computer
PCD	Policy controlled Cognitive Devices
PCR	Policy Conformance Reasoner
PDL	Policy Description Language
PDP	Policy Decision Point
PE	Policy Engine
PED	Policy Engine Database
PEM	Policy Engine Manager
PEP	Policy Enforcing Point
PHT	Policy Handling Toolbox
PHY	Physical layer
PI	Policy Interface
PL	Policy Language
PLL	Phase-Locked Loop
PM	Policy Manager
PMP	Policy Management Point
PoA	Point of Attachment
PR	Policy Reasoner
PS	Policy Server
PSD	Policy Server Database
PSDH	Policy Server Database Handler

QoS	Quality of Service
RAN	Radio Access Networks
RAT	Radio Access Technology
RCP	Raw Context Processing
RDF	Resource Description Framework
RE	Radio Environment
REng	Reasoning Engine
RF	Radio Frequency
RRA	Radio Resource Assignment
RRM	Radio Resource Management
RTP	Real-time Transport Protocol
SCH	Superframe Control Header
SD	Secure Digital
SDR	Software Defined Radio
SM	Spectrum Manager
SON	Self Organizing Network
SSA	Spectrum Sensing Automation
SSF	Spectrum Sensing Function
SSRC	System Strategy Reasoning Capability
SU	Secondary User
TCP	Transmission Control Protocol
TDMA	Time Division Multiple Access
TI	Texas Instruments
TRM	Terminal Reconfiguration Manager
TV	Television
UDP	User Datagram Protocol
USB	Universal Serial Bus
USRP	Universal Software Radio Peripheral
WLAN	Wireless Local Area Network
WP	Work Package
WRAN	Wireless Regional Area Network
XG	neXt Generation
XGPL	neXt Generation Policy Language
XML	Extensible Markup Language

7. References

[1] D. Denkovski, V. Pavlovska, V. Atanasovski and L. Gavrilovska, "Novel Policy Reasoning Architecture for Cognitive Radio Environments", in *IEEE Global Telecommunications Conference (GLOBECOM),* Miami, FL, 2010, pp. 1-5.

[2] EC FP7-216856 ARAGORN Project, Deliverable D3.3, "Final System Architecture", May 2010.

[3] XG Prolog Policy Engine. Available at: http://xg.csl.sri.com/prolog.php.

[4] XG Working Group, "The XG Vision, Request for Comments, Version 2.0", tech. report, BBN Technologies, 2005.

[5] D. Elenius, G. Denker, M.-O. Stehr, R. Senanayake, C. Talcott and D. Wilkins, "CoRaL–Policy Language and Reasoning Techniques for Spectrum Policies," in *8th IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY)*, Bologna, 2007, pp. 261-265.

[6] Universal Software Radio Peripheral 2 (USRP2). Information available at: http://www.ettus.com.

[7] EC FP7-248303 QUASAR Project, Deliverable D4.1, "Sharing strategies for unaware secondary systems", March 2010.

[8] D. Denkovski, V. Atanasovski and L. Gavrilovska, "Policy enforced spectrum sharing for unaware secondary systems", in *4th International Conference on Cognitive Radio and Advanced Spectrum Management (CogART)*, Barcelona, 2011.

[9] L. Gavrilovska and V. Atanasovski, "Spectrum Sensing Framework for Cognitive Radio Networks", *Springer Wireless Personal Communications*, *"Special Issue: Cognitive Networks and Spectrum Management (Selected Topics from the CTIF Workshop, May 31–June 1, 2010, Aalborg University, Denmark)"*, vol. 59, no. 3, pp. 447-469, 2011.

[10] FP6-IST-027714 E2R Project, End-to-End Reconfigurability.

[11] ICT-2007-216248 E3 Project, End-to-End Efficiency. Available at: https://ict-e3.eu/.

[12] *IEEE Standard for Architectural Building Blocks Enabling Network-Device Distributed Decision Making for Optimized Radio Resource Usage in Heterogeneous Wireless Access Networks*, IEEE Std 1900.4TM-2009, Jan. 2009.

[13] S. Buljore, H. Harada, P. Houze, K. Tsagkaris, O. Holland, S. Filin, T. Farnham, K. Nolte and V. Ivanov, "Architecture and Enablers for Optimised Radio Resource usage: The IEEE P1900.4 Working Group", *IEEE Communications Magazine*, vol. 47, no. 1, pp. 122-129, Jan. 2009.

[14] ICT-2007-216248 E3 Project, Deliverable D4.8, "Empirical feasibility evaluations of autonomous functionalities", Dec. 2009.

[15] ICT-2007-216248 E3 Project, Deliverable D4.4, "Final solution description for autonomous CR functionalities", Sept. 2009.

[16] ICT-2007-216248 E3 Project, Deliverable D4.5, "Final system specification for autonomous CR functions", Dec. 2009.

[17] ICT-2007-216248 E3 Project, Deliverable D4.7, "Final performance and complexity analysis for autonomous CR functionalities", Sept. 2009.

[18] A. Galani, K. Tsagkaris, N. Koutsouris and P. Demestichas, "Design and assessment of functional architecture for optimized spectrum and radio resource management in heterogeneous wireless networks", *Wiley International Journal of Network Management*, vol. 20, no. 4, pp. 219–241, July/Aug. 2010.

[19] IST-2004 027965 ORACLE Project, Deliverable D4.1, "Draft OR Policy Framework", Nov. 2006.

[20] IST-2004 027965 ORACLE Project, Deliverable D4.2, "Definition of context filtering mechanisms and policy framework", May 2007.

[21] IST-2004 027965 ORACLE Project, Deliverable D4.3, "OR decision making engine definition", April 2008.

[22] IST-2004 027965 ORACLE Project, Opportunistic Radio Communications in unLicensed Environments.

[23] DARPA XG Working Group, "DARPA XG Policy Language Framework, Request for Comments", v. 1.0, prepared by BBN Technologies, Cambridge MA, USA, 2004.

[24] IST-2004 027965 ORACLE Project, Deliverable D4.4, "OR policy based decision engine demonstrator", Oct. 2008.

[25] *"IEEE Standard for Information Technology--Telecommunications and information exchange between systems Wireless Regional Area Networks (WRAN)--Specific requirements Part 22: Cognitive Wireless RAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Policies and Procedures for Operation in the TV Bands"*, IEEE Std 802.22-2011, pp.1-680, July 1 2011.

[26] M.M. Kokar and L. Lechowicz, "Language Issues for Cognitive Radio", *Proceedings of the IEEE*, vol.97, no.4, pp.689-707, April 2009.

[27] M. Cummings and P.A. Subrahmanyam, "Perspectives of a metalanguage for configurable wireless systems", in *Software Defined Radio Technical Conference (SDR.'04)*, 2004.

[28] G. Denker, D. Elenius, R. Senanayake, M.-O. Stehr, C. Talcott and D. Wilkins, "Cognitive Policy Radio Language (CoRaL) A Language for Spectrum Policies XG Policy Language", Version 0.1, SRI Project No. 16763, April 1, 2007.

[29] D. Wilkins, G. Denker, M-O. Stehr, D. Elenius, R. Senanayake and C. Talcott, "Policy-Based Cognitive Radios", *IEEE Wireless Communications*, vol.14, no.4, pp.41-46, Aug. 2007.

[30] *"IEEE Standard for Policy Language Requirements and System Architectures for Dynamic Spectrum Access Systems"*, IEEE Std 1900.5™-2011, January 13 2012.

[31] L. Kagal, T.Finin and A. Joshi, "A Policy Language for a Pervasive Computing Environment", in *IEEE 4th International Workshop Policies for Distributed Systems and Networks (POLICY)*, 2003, pp. 63-74.

[32] A. Uszok, J.M. Bradshaw, M. Johnson, R. Jeffers, A. Tate, J. Dalton and S. Aitken, "KaoS Policy management for Semantic Web Services", *IEEE Intelligent Systems*, vol. 19, issue 4, pp. 32-41, July-Aug. 2004.

[33] C Language Integrated Production System, http://www.ghg.net/clips/CLIPS.html, June, 2006.

[34] J. Lobo, R. Bhatia and S. Naqvi, "A policy description language". in *16th National Conference on Artificial Intelligence*, Orlando, FL, July 1999.

[35] N. Damianou, N. Dulay, E. Lupu and M. Sloman, "The Ponder Policy Specification Language", in *International Workshop on Policies for Distributed Systems and Networks (POLICY)*, 2001, pp. 18-38.

[36] IEEE DySPAN Standards Committee, http://www.dyspan-sc.org/.

[37] P. Cordier, P. Houze, S.B. Jemaa, O. Simon, D. Bourse, D. Grandblaise, K. Moessner, J. Luo, C. Kloeck, K. Tsagkaris, R. Agusti, N. Olaziregi, Z. Boufidis, E. Buracchini, P. Goria and A. Trogolo, "E2R cognitive pilot channel concept", in *15th IST Mobile and Wireless Communications Summit (IST)*, Mykonos, 2006.

[38] ICT- 257626 ACROPOLIS Project, Deliverable 12.1, "Specification of Preliminary Set of Appropriate Metrics, Utility Functions and Layer Identification", September 2011.

[39] G. Baldini, V. Rakovic, V. Atanasovski and L. Gavrilovska, "Security aspects of policy controlled cognitive radio", in *5th IFIP International Conference on New Technologies, Mobility & Security (NTMS)*, Istanbul 2012.

[40] V. Pavlovska, D. Denkovski, V. Atanasovski and L. Gavrilovska, "A Policy Reasoning Architecture for Cognitive Radio Networks", in *8th International Conference on Communications (COMM)*, Bucharest, 2010.

[41] S. Gupta, P. Murphy, C. Hunter and A. Sabharwal, "Cognitive Radio Platforms: Wireless Open-Access Research Platform for Networks", Rice University.

[42] K.R. Chowdhury and T. Melodia, “Platforms and Testbeds for Experimental Evaluation of Cognitive Ad Hoc Networks”, *IEEE Communications Magazine*, vol. 48, no. 9, pp. 96-104, Sept. 2010.

[43] ICT-248351 Faramir Project, Deliverable D2.1, “State of the Art”, April 2010.

[44] L.A. DaSilva, A.B. MacKenzie, C. da Silva and R.W. Thomas, “Requirements of an Open Platform for Cognitive Networks Experiments”, in *3rd IEEE Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, Chicago, IL, 2008, pp. 1-8.

[45] *“IEEE Standard for Spectrum Sensing Interfaces and Data Structures for Dynamic Spectrum Access and other Advanced Radio Communication Systems”*, IEEE Std 1900.6-2011, April 22 2011.

[46] S. Koslowski, M. Braun, J.P. Elsner and F.K. Jondral, “Wireless Networks In-the-Loop: Emulating an RF front-end in GNU Radio”, in *SDR Forum 2010 European Reconfigurable Radio Technologies Workshop*, Mainz, 2010.

[47] ICT- 257626 ACROPOLIS Project, Deliverable D7.1, “Evaluation and Roadmap of Existing SDR, CR Platforms and Future CR Systems”, September 2011.

[48] GNU Radio software development toolkit. Information available at: http://gnuradio.org/redmine/wiki/gnuradio.

[49] LabView NI-USRP drivers support. Information available at: http://joule.ni.com/nidu/cds/view/p/id/2679/lang/en.

[50] USRP SDR support by Matlab Simulink. Information available at: http://www.mathworks.com/discovery/sdr/usrp.htm.