A POLICY STRATEGY FOR THE SOFTWARE DEFINED RADIO ENABLING COGNITIVE RADIO TECHNOLOGY WITHIN THE TACTICAL NETWORK

Lynn Grande
General Dynamics C4 Systems
Taunton, MA
lynn.grande@gdc4s.com

Matt Menard
General Dynamics C4 Systems
Taunton, MA
matt.menard@gdc4s.com

Communicating mission goals to the Software Defined Radio in the tactical network requires a policy driven approach. The bulk of policy distribution is currently a manual, paper intensive process. Creating a more automated policy strategy brings the Software Defined Radio a step closer to creating the Cognitive Radio layer. This strategy includes policy management, decision making, enforcement and distribution. Definition of this policy strategy shall include standard policy models developed by the DMTF, facets of reasoning technologies in the decision processes and Object Oriented device models. This paper will address policy issues and solutions that enable the next step toward Cognitive Radio.

Mission requirements dictate the movement of platforms, devices and users, as well as security needs, and the management of frequencies to prevent interference and spectrum policy violation. Communications equipment that can dynamically analyze, understand and adapt to the constantly changing environment to maintain the connectivity and communication links are vital to the mission success. Cognitive radios will have the ability to monitor the tactical network environment and recognize situations that require changes in operation. Policies can be used to invoke changes in radio operation based on specific network conditions. The bi-directional nature of policy based services support the need for dynamic adaptability. Changes in environment will trigger the cognitive radio to retrieve a policy and transparently migrate to a new configuration, based on the policy definition. The result of which is that the configuration and the operation of the radio changes in an effort to maintain communication links.

One significant issue is developing a policy definition scheme for disparate networks that allows for policies to be applied to diverse types of network equipment. [1] By employing the use of an object-based policy definition, a consistent policy structure can be developed. The benefit to an object-based policy definition is a consistent and flexible policy definition where all network resources are defined in terms of an object structure. This enables network devices to have a similar subset of attributes and leaves the actual policy implementation to the individual network device

Another issue wrought with technical challenges is the area of policy dissemination. How can policies be distributed from the policy production point to all of the policy consumers? There are a number of different answers to this question. The most popular methods of policy distribution are using typical data persistence mechanisms, such as databases, directories and file systems. Of course, each of these mechanisms has its own benefits and drawbacks. Depending on individual requirements one may be a better fit than the others.

Policy Architecture

The key to creating a flexible and usable policy implementation is to start with standards based policy architecture and data model. Policy architecture is described in a variety of models; the most common model is heavily influenced by the CIM-based IETF (Internet Engineering Task Force) Policy Core Information Model (PCIM), which is defined by the IETF Policy Framework Working Group [RFC3060, RFC3460][2]. Figure 1 illustrates this policy framework as described in the Policy Core Information Model consists of four basic elements:

1. A policy management tool;
2. A policy repository;
3. A policy decision point; and
4. A policy enforcement point.

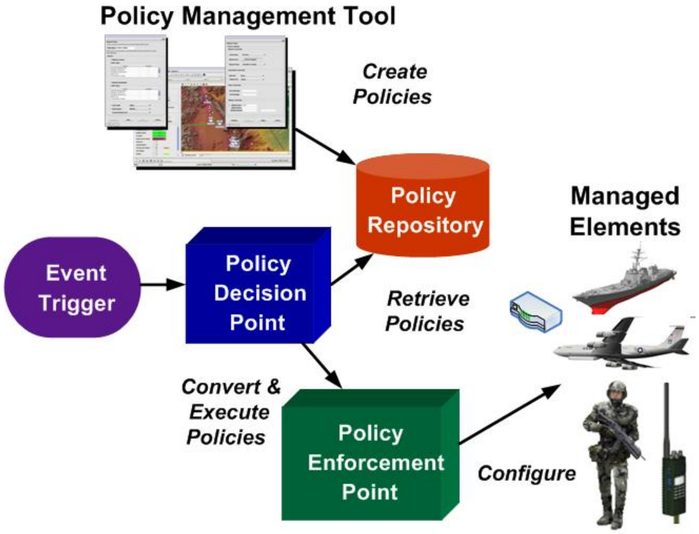


Figure 1 – PCIM Policy Framework

The *policy management tool* is used to input active policies. This tool takes high-level policy information and constructs a more detailed, low-level policy description that can be applied to various devices in the network. This should be implemented as an intuitive graphical user interface (GUI) and can be either a standalone application or a component of a mission/network planning system. The resulting detailed policy description is stored in the *policy repository*.

The *policy repository* is most commonly a database, files or directory service. When persisted in a directory service, policy distribution becomes a part of the overall directory replication mechanism. This enables the policy descriptions to be automatically distributed throughout the network and aids in the synchronization of policies between the creation points and the enforcement and decision points.

Policy enforcement points (PEP) enforce and execute the different policies, and can use intermediary *policy decision points (PDP)* instead of communicating directly with the repository. When an event in the system indicates that a policy must be applied, the PDP takes the event trigger and retrieves the policy information from the repository. This information may be converted to a format the PEP understands, in most cases, the conversion is from the general policy language to a specific device configuration. The PEP then executes the policy. Some policy architectures combine the PDP and PEP functions. Common examples of policy enforcement points are routers, firewalls and gateways.

Policy Information Model

DMTF standards provide common management infrastructure components for instrumentation, control and communication that are technology and platform independent [3], and represented by the Common Information Model (CIM). The goal of the CIM is to model all areas of a managed environment, such as networks, users, equipment, policies and applications. Policy is defined within the CIM as a set of rules to administer, manage, and control access to network resources. [4] The CIM Policy Model emphasizes the definition of general event-condition-action semantics. These semantics are represented abstractly, independent of any policy language or implementation. The three main classes to support the policy semantics are the PolicyRule, PolicyCondition and PolicyAction. The PolicyRule is comprised of PolicyConditions and PolicyActions, which follows the "IF PolicyCondition [*policy variable EQUALS policy value*] THEN EXECUTE PolicyAction [*SET policy variable TO policy value*]". Figure 2 illustrates the basic classes of the CIM Policy Model.

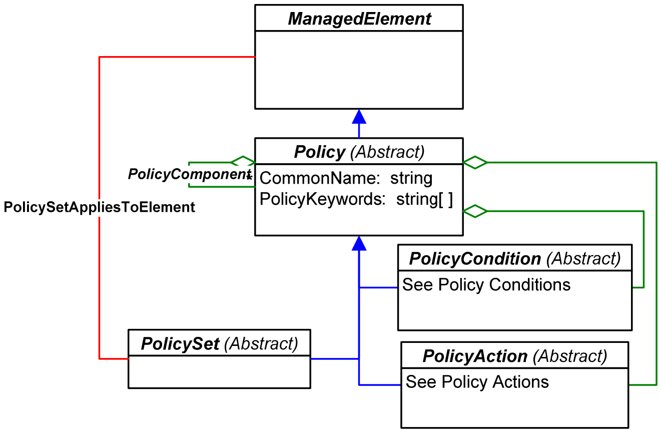


Figure 2 Basic Classes of CIM Policy

Policy Dissemination

Policy dissemination is one of the more challenging areas of the policy generation cycle. Currently the three most commonly used methods of policy persistence and dissemination involve databases, directories and files. Each has specific benefits and drawbacks. Directory services have a unique advantage in that they have a built in replication scheme that enables data updates to be distributed throughout a network in near real-time. However, directories are more suited data with low volatility and the replication can consume a large amount of bandwidth. Therefore, a directory may not be suited to a system with high data volatility; likewise a directory system would not be a prime candidate for a system with limited or unreliable bandwidth.

Databases, on the other hand, are much more suited for more volatile data and typically have much better performance than a directory service. One issue, with a database, is data synchronization. While some databases do have a synchronization scheme the updates are performed much less frequently, only certain kinds of data can be synchronized and the synchronization scheme is much less robust than that of a directory service.

One other way to store policy descriptions is in a file structure (e.g.: XML, flat-files, etc.). In most scenarios, files are accessed by a single process and therefore data volatility is not a major concern. However, distributing the policy files to various points within the network can be logistically challenging and time consuming.

These are, of course, high-level overviews of some of the various dissemination schemes and there are others, however most use one of these schemes as a starting point and build from it.

Applicability to Cognitive Radio

The cognitive radio starts with Software Defined Radio (SDR) technology and adds a layer of artificial intelligence that senses the radio's environment and adapts to it. Policy just provides some of the intelligence rules and mission guidance for the cognitive engine.

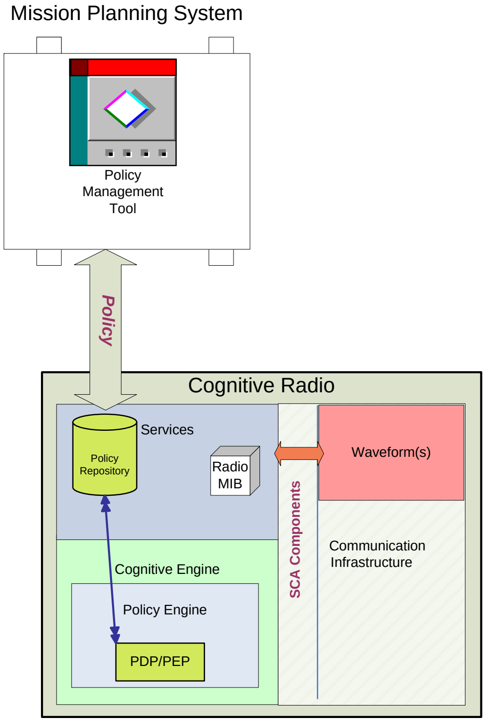
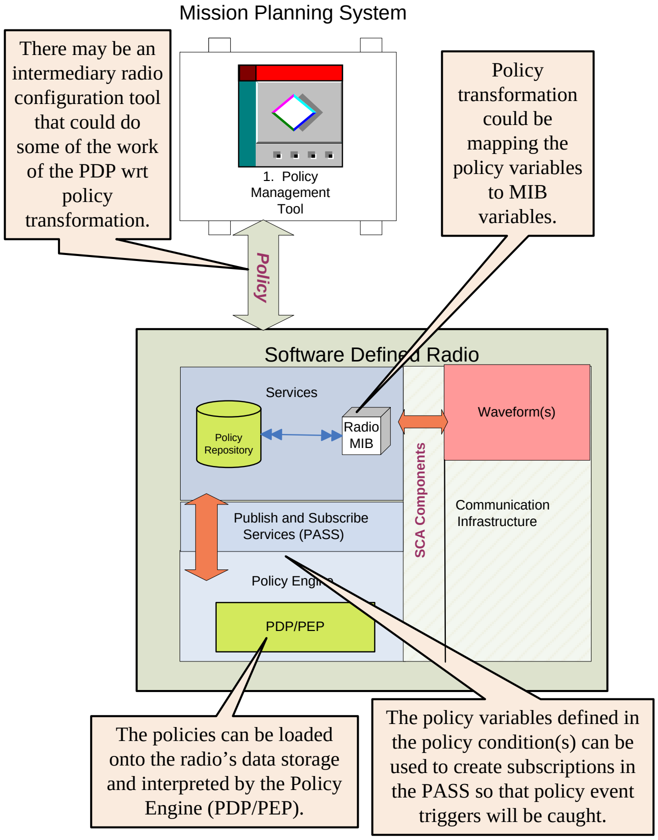


Figure 3 – Limited Logical View of Policy within CR

Figure 3 shows a limited logical view of policy implementation within the Cognitive Radio. The policy engine is made up of the policy decision point and enforcement point which is a subset of the cognitive reasoning engine.

The first step is to utilize the information that is available in today's tactical networks. Most devices have the capability to provide information about itself. Generally, the software defined radio and many other devices will have a Management Information Base (MIB) allowing configuration changes and information retrieval. SNMP is a common protocol used to access the MIB. A MIB object reference accesses data persisted in the device. Most devices also implement many of the standard networking MIBs as part of the MIB II [5] specification. This gives us the basis for some simple software intelligence for implementing policies. Using SNMP, a common protocol used to access MIBs, the radio can report its status or receive updates to alter its configuration and operation.

Figure 4 illustrates a potential policy implementation in the current SDR technology.



There may be an intermediary radio configuration tool that could do some of the work of the PDP wrt policy transformation.

Policy transformation could be mapping the policy variables to MIB variables.

The policies can be loaded onto the radio's data storage and interpreted by the Policy Engine (PDP/PEP).

The policy variables defined in the policy condition(s) can be used to create subscriptions in the PASS so that policy event triggers will be caught.

Figure 4 – Potential Policy Implementation in SDR

One of the PDP functions would be to map the policy variables in the conditions and actions to the radio MIB variables. The PDP function can be implemented in such a way to use the publish and subscribe paradigm for subscribing to and receiving event triggers from the MIB. When an event occurs the condition can be fully evaluated and a decision can be made to execute one or more actions.

Another approach would be to use an intermediary host platform incorporating the PDP/PEP. The PDP component would be responsible reacting to trigger events in the network and retrieving policy descriptions from the repository based on the events. Using the policy descriptions, the PDP would create a set of modifications to the radio's MIB variables, which would be sent to the PEP. The PEP component would then be responsible for altering the radio's configuration by using SNMP to modify the MIB, which, in turn, would trigger changes in the radio's operation.

Policy XML

In order to transfer the policies between planning systems, host platforms, radios and devices would be to create an XML schema based on the PCIM. For ease of transformation between vendors it is best that the policy variables are based on an object oriented information model. So extending the use of the Common Information Model (CIM), the policy variables would align with objects and attributes defined in the CIM. Figure 5 illustrates a sample portion of a cognitive radio policy schema.

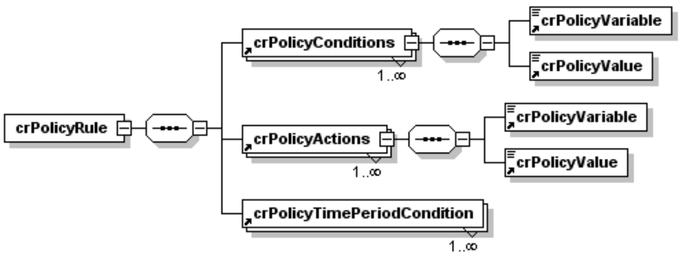
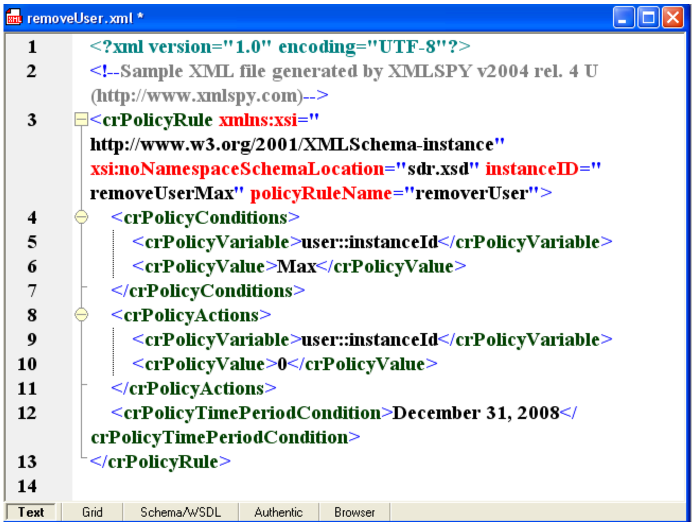


Figure 5 – Sample Cognitive Radio Policy XML

The policy can be exported from the mission planning system as XML. The XML file(s) can then be transported to the PDP/PEP via SOAP or any network file transfer mechanism.

User account management is one example of a mission function that can be performed through the use of policy. For example, the sample policy XML file in figure 6 defines an action that is to remove user "Max" on December 31, 2008. Examining the policy XML, for this simplistic example, note that the policy condition is that there is an account with an instance ID of "Max". Additionally, there is a date constraint, which indicates that this policy is not valid until a specific date, in this case December 31, 2008. The policy action, which is enacted when the conditions occur, is that the user's instance ID be modified from "Max" to "0."

This XML file would be given to the PDP and on December 31, 2008 the PDP would act on this policy and alert the PEP that the account for user Max is to be deleted. It is now the responsibility of the PEP to delete the user account by setting user "Max's" instance ID to 0.



```
<?xml version="1.0" encoding="UTF-8"?>
<!--Sample XML file generated by XMLSPY v2004 rel. 4 U
(http://www.xmlspy.com)-->
<crPolicyRule xmlns:xsi="
http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="sdr.xsd" instanceID="
removeUserMax" policyRuleName="removerUser">
    <crPolicyConditions>
        <crPolicyVariable>user::instanceId</crPolicyVariable>
        <crPolicyValue>Max</crPolicyValue>
    </crPolicyConditions>
    <crPolicyActions>
        <crPolicyVariable>user::instanceId</crPolicyVariable>
        <crPolicyValue>0</crPolicyValue>
    </crPolicyActions>
    <crPolicyTimePeriodCondition>December 31, 2008</
crPolicyTimePeriodCondition>
</crPolicyRule>
```

Figure 6 - Sample XML Policy Rule

Relationship Between Policy Engine and Cognitive Engine

The implementation scheme described above is not intended to replace a full policy engine. It is only an intermediary step in CR policy implementation. A policy engine using one of the standard policy languages (PONDER, XACML, CIM-SPL, etc) would be more scalable to a full set of network devices. Since the policy engine is a subset of the overall cognitive engine, it seems logical to map the policy language to the ontology language used for the cognitive engine. There is a great deal of research proposing these mappings. Stephen Quirolgico [6], *et al* proposed a framework for constructing a CIM ontology based upon previous research that identified mappings from Unified Modeling Language (UML) constructs to ontology language constructs. Jorge E. López de Vergara [7], *et al* presents an XML based ontology language that maps Web Ontology Language (OWL) constructs to CIM elements.

Conclusion

Software defined radios are more flexible and adaptive to tactical environments than the traditional radios that preceded them. This benefits the soldier by allowing one radio to operate various waveforms and essentially become multiple radios in one package. Enabling software defined radios with policy architecture is one more step down the path to a true cognitive radio. Policy-enabled radios will further benefit the soldier by disconnecting the human aspect of monitoring the tactical communication environment and push that responsibility onto the radio itself. With this technology, the radio can implement and enforce policies based on the ever changing tactical network environment conditions. The result is a radio that can seamlessly and transparently maintain communications links in an ad-hoc mobile tactical environment.

10. REFERENCES

[1] Reuben S. Fischman, Adam T. Payne, POLICY BASED NETWORK OPERATIONS AND MANAGEMENT FOR TRANSFORMATIONAL NETWORKS, Proceedings of MILCOM 2007 Orlando, Fl.

[2] From CIM Policy Model White Paper www.dmtf.org/standards/documents/CIM/DSP0108.pdf

[3] From Distributed Management Task Force, Inc. www.dmtf.org/about .

[4] Policy Core Information Model [RFC3060] http://www.ietf.org/rfc/rfc3060.txt

[5] [Management Information Base for Network Management of TCP/IP-based internets: MIB-II [RFC1213] http://www.ietf.org/rfc/rfc1213.txt.

[6] "Toward a Formal Common Information Model Ontology" Stephen Quirolgico, Pedro Assis, AndreaWesterinen, Michael Baskey, and Ellen Stokes; http://w3.antd.nist.gov/pubs/quirolgico-wise2004.pdf .

[7] ""Applying the Web Ontology Language to Management Information Definitions",
Jorge E. López de Vergara, Víctor A. Villagrá, and Julio Berrocal; http://jungla.dit.upm.es/~jlopez/publicaciones/ieeecomm04.pdf .

Copyright Transfer Agreement: The following Copyright Transfer Agreement must be included on the cover sheet for the paper (either email or fax)—not on the paper itself.

“The authors represent that the work is original and they are the author or authors of the work, except for material quoted and referenced as text passages. Authors acknowledge that they are willing to transfer the copyright of the abstract and the completed paper to the SDR Forum for purposes of publication in the SDR Forum Conference Proceedings, on associated CD ROMS, on SDR Forum Web pages, and compilations and derivative works related to this conference, should the paper be accepted for the conference. Authors are permitted to reproduce their work, and to reuse material in whole or in part from their work; for derivative works, however, such authors may not grant third party requests for reprints or republishing.”

Government employees whose work is not subject to copyright should so certify. For work performed under a U.S. Government contract, the U.S. Government has royalty-free permission to reproduce the author's work for official U.S. Government purposes.