

A Reconfigurable Platform for Cognitive Networks

Paul Sutton
CTVR
Trinity College Dublin
suttonpd@tcd.ie *

Linda E. Doyle
CTVR
Trinity College Dublin
ledoyle@tcd.ie

Keith E. Nolan
CTVR
Trinity College Dublin
keith.nolan@ctvr.ie

Abstract

By introducing self-awareness and computational intelligence to reconfigurable radio networks, cognitive networks are viewed as the key to a new generation of self-configuring, self-optimizing and self-healing communications systems. However, in order to make such systems realizable, a paradigm shift in the design of network node architectures is required. This paper presents a reconfigurable platform based on an architecture specifically designed for nodes within a cognitive network.

1. Introduction

Cognitive networks hold the potential to provide robust, high-bandwidth, adaptive communications systems through their ability to observe the current network conditions, reason about their environments and adapt to use the resources available in the most efficient manner possible.

Today's networks are typically comprised of nodes within which the ability to observe, reason and adapt is restricted to the individual elements which make up their structure. This can be seen particularly in networks of nodes which adopt a layered protocol architecture. Although the use of self-contained protocol layers facilitates the use of abstraction to simplify the design of highly complex systems, it also precludes the ability of the node to reason across observations made at a number of layers and adapt in a holistic manner.

In this paper, we present a node architecture specifically designed for cognitive networks to address these deficiencies and facilitate the requirements of network-wide observation and adaptation.

Section 2 defines the principal functions of the reconfigurable node architecture using the concept of the cognition

cycle. Section 3 outlines the approach taken in the reconfigurable node design and presents the key elements and mechanisms implemented. Section 4 concludes the paper.

2. Background

Mitola first described the operation of a cognitive radio in terms of a feedback loop [4], where he outlined the concept of the cognition cycle. The cognition cycle attempts to model the manner in which a cognitive system interacts with its environment and assimilates knowledge which is used to guide decisions affecting its performance. Using the cognition cycle, Mitola identifies six processes which together allow a cognitive system to '*employ model-based reasoning to achieve a specified level of competence in radio-related domains*' [3]. These processes are:

1. Observing the outside world.
2. Orientation of the system.
3. Planning one or more courses of action.
4. Deciding upon a course of action.
5. Acting to influence the operation of the system.
6. Learning from experience.

The process of *observing* the outside world involves the acquisition of knowledge by a cognitive radio about its environment through the analysis of incoming information streams. The *orientation* process of the cycle is concerned with establishing the priority attached to any observations which are made. Certain observations may require immediate action while others may feed into other cycle processes such as the *planning* process. This process is responsible for generating and analyzing courses of action which may be taken. At the *decision* stage of the cycle, one of these courses is chosen. Finally the decision is put into *action* and the operation of the cognitive radio is actually influenced.

*This material is based upon work supported by Science Foundation Ireland under Grant No. 03/CE3/I405 as part of the Centre for Telecommunications Value-Chain Research (CTVR) at Trinity College Dublin, Ireland.

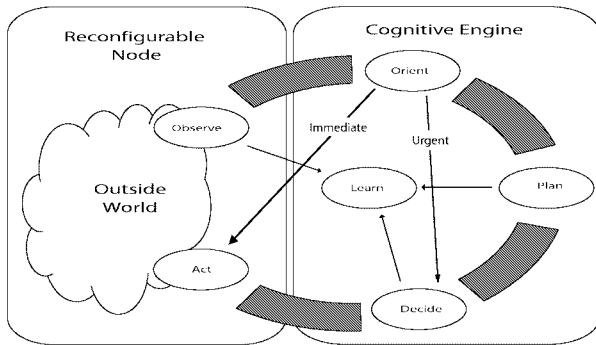


Figure 1. The Cognition Cycle

The *learning* process allows the system to learn from experience. Similar cycles are used to describe the operation of cognitive radios by Haykin [1] and Thomas [6].

A simplified reproduction of the cognition cycle is shown in Figure 1. From our perspective, the cognition cycle can be divided into two entities as illustrated. The first entity which we have termed the *Cognitive Engine* concerns itself broadly with reasoning, cognition and deduction. The second entity is focussed solely on the processes of observation and action. We have termed this entity the *Reconfigurable Node*.

The Reconfigurable Node forms a platform for cognitive networks by providing an architecture designed specifically for reconfiguration and observation, not only at the radio layers of the individual node, but also throughout the node network stack and on a network-wide scale.

3. The Reconfigurable Node Design

The challenge therefore is to design a node of a network that is highly flexible and adaptable in order to be able to:

1. Make the network and node level observations required by the Cognitive Engine.
2. Respond to instructions and reconfigure all aspects of the node operation as necessary.

One way to approach the design of such a Reconfigurable Node is to view a node in the network as a structure comprised of heterogeneous hardware and software components. These components are linked together to form a desired node configuration and may be dynamically altered as desired. Any part of a node may be represented as an individual component and these components may have greatly varying levels of granularity. For example, a component may encapsulate an entire node layer (such as the Network or Physical layer of the OSI model) within a network stack architecture or may contain a subsection of such

a layer (such as a filter or digital modulator). Each component will have a number of associated parameters which control its operation (such as the cache timeout of an ad-hoc routing layer or the tap values of a filter). A significant advantage of adopting such a component-based design is the ability to manage the high degree of complexity required in a system such as the Reconfigurable Node. By encapsulating the implementation of specific parts of the node in components, it is possible for a node designer to effectively build a wide range of complex systems without detailed knowledge of every aspect of each component used. This may be achieved through the use of a universal component container or shell which is used to wrap the unique functionality of each heterogeneous component and allow it to be handled in a generic manner. The use of components also permits the creation of extensible, flexible systems and enables the reuse and adaptation of specific components in a number of differing system configurations. A thorough examination of the principles of component-based design can be found in [5].

The following sections first outline the principal components of the Reconfigurable Node design, then present the key elements responsible for the management and control of components within the framework, before finally examining the mechanisms provided to perform the core functions of observation and action outlined in section 2.

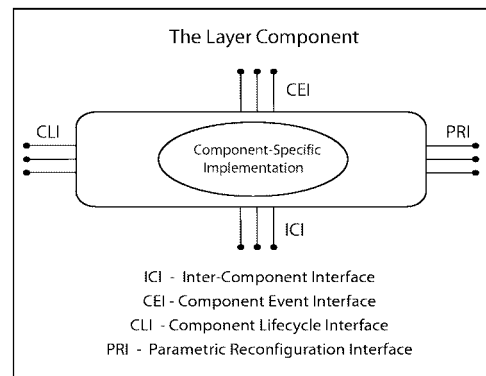


Figure 2. The Layer Component

3.1. Components

Two specific component types make up the fundamental building blocks of the Reconfigurable Node. These are the Layer Component and the Radio Component.

The *Layer Component* (LC) is designed to encapsulate the functionality of a layer within a network stack such as the network layer of an OSI model-based architecture. As can be seen in Figure 2, the LC provides four principal interfaces to allow each layer of the network stack to be handled

in a generic manner. The *Inter-Component Interface* permits components to be linked together to form a network stack structure and facilitates the flow of data upwards and downwards over this link. The *Component Lifecycle Interface* is provided to allow an external controller to start, suspend, resume and stop the operation of the component as well as initialize and destroy the component itself. The *Parametric Reconfiguration Interface* facilitates the manipulation of a component's operating parameters in a generic manner and the *Component Event Interface* is used to notify an external listener of any significant occurrences within the layer.

The second type of component used in the Reconfigurable Node is the *Radio Component (RC)*. IRIS (Implementing Radio In Software) [2] is a highly flexible and reconfigurable software radio architecture which runs on a general-purpose processor and has been adapted for use as a physical layer component within the reconfigurable node architecture. The Radio Component is the basic building block of the IRIS system and is used to encapsulate a single signal processing step of an isochronous signal chain. The use of IRIS at the physical layer of the Reconfigurable Node allows the broad range of parameters associated with the physical layer of a wireless node to be monitored and reconfigured. Although the RC provides a number of interfaces which correspond to those of the Layer Component, the implementation details of the two are very different. Throughout the remainder of this paper, the term *Component* is used to refer to both.

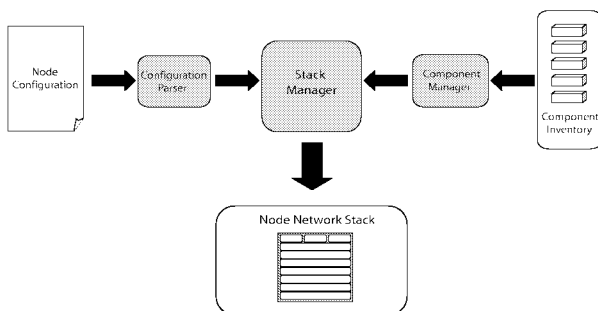


Figure 3. The Management Elements

3.2. Framework

Figure 3 illustrates the principal elements of the framework involved in the process of creating and managing the node network stack. These elements are the Stack Manager, the Configuration Parser and the Component Manager, each of which is presented in turn below.

The key controlling entity of the reconfigurable node is the *Stack Manager (SM)*. Using the layer and radio components as the basic building blocks, the SM constructs the

network stack of the node and controls its operation. Given a stack configuration, the SM identifies, creates and initializes the required components. The components are then linked together to form the structure of the node's network stack and the operation of each is started. Throughout the operating period of the network stack, the SM is responsible for handling and reacting to any events signalled by the components as well as performing any reconfiguration required. When the operation of the network stack ceases, the SM halts the operation of each component, unlinks them to break up the stack structure and destroys them.

In order to carry out these tasks successfully, the Stack Manager makes use of two additional features of the reconfigurable node framework - the *Configuration Parser* and the *Component Manager*.

A fundamental requirement of the Reconfigurable Node is the ability to specify the initial parameter settings of a number of components, as well as how those components should be linked together to form the network stack. The *Configuration Parser (CP)* is the element of the reconfigurable radio with the responsibility of translating the network stack configuration into a lightweight data structure which can be read by the Stack Manager. It was decided to use the XML open standard [8] to specify the node network stack configuration. XML allows hierarchical data to be represented in a concise and unambiguous manner while also providing the flexibility to describe any potential component parameters and capture the structure of network stacks with a high degree of complexity. An XML parser [7] is used to validate configuration documents against an XML schema and to represent these documents as tree structures which may be navigated. With the XML parser as its engine, the CP creates a lightweight data structure from a configuration document. This data structure is then used by the Stack Manager to create and initialize components and link them together to form the node network stack. As the state of the network stack changes during the course of its operation, these alterations are reflected in the data structure. The CP also provides the ability to create a full XML configuration document from the stack data structure and thus capture the stack state at any time.

The second feature of the Reconfigurable Node which acts to provide a number of services to the Stack Manager is the *Component Manager (CM)*. The Component Manager maintains an inventory of components which may be used by the SM to build the node network stack. These components may reside locally or on one or more remote networked inventories. As components are requested by the Stack Manager, they are retrieved from an inventory, created and made available. When an existing component is no longer required, the Component Manager is responsible for its destruction.

Together with the Configuration Parser and the Compo-

nent Manager, the Stack Manager permits the network stack of a node to be specified using an XML configuration document and dynamically built using a number of heterogeneous hardware and software components. However, in order to perform the key roles of observing the outside world and altering its operation in response to external instructions defined in section 2, a number of additional mechanisms are required. These mechanisms are illustrated in Figure 4 and described in the following sections.

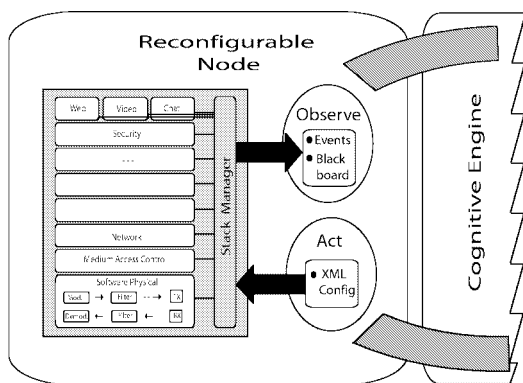


Figure 4. The Reconfigurable Node and the Cognition Cycle

3.3. Observation

As discussed in section 2, one of the two fundamental roles to be played by the Reconfigurable Node is that of observing the outside world and making these observations available to be used by the Cognitive Engine.

To address this challenge, two mechanisms for fulfilling this role are provided. Together these mechanisms permit observations to be both pushed to the Cognitive Engine by the Reconfigurable Node and pulled from the Reconfigurable Node by the Cognitive Engine.

The first of these mechanisms allows any component within the node network stack to notify the Stack Manager of a significant occurrence. These *Events* may be defined by the designer of the individual component and can be used to highlight any occurrences that the designer considers of use to the Cognitive Engine. Examples include the crossing of a given threshold for the bit error rate within a Reed-Solomon decoder component or for the link change rate of an ad-hoc routing component. By signalling a given event, the component makes the Stack Manager aware of the specific occurrence and allows it in turn to inform the Cognitive Engine.

The second mechanism, referred to above, permits observations of the outside world to be pulled from the Reconfigurable Node by the Cognitive Engine. This mechanism

consists of a shared database referred to as the Reconfigurable Node *Blackboard*. The Blackboard permits each of the components of the network stack to expose any information which may be used to inform decisions made by the Cognitive Engine. Examples of such information could be the power spectral density obtained at the FFT stage of an OFDM receiver or the routing table of a network layer routing protocol.

By providing both a push and pull mechanism for the exchange of observations between the Reconfigurable Node and the Cognitive Engine, both reactive and proactive cognitive radio approaches may be facilitated.

3.4. Action

The second fundamental role to be played by the Reconfigurable Node is that of permitting every aspect of its operation to be dynamically altered in response to external instructions. It is this flexibility that reflects the action process of the cognition cycle referred to in section 2.

In discussing node reconfiguration, it is useful to categorize a number of different levels or degrees of reconfiguration which may occur.

The following levels have been identified:

1. Parametric reconfiguration.
2. Structural reconfiguration.
3. Application reconfiguration.

Parametric reconfiguration involves the dynamic alteration of specific parameters within individual components of the node network stack. Examples include the alteration of a single tap value in a signal filter component or of the timeslot length in a TDMA MAC layer component.

Structural reconfiguration refers to changing individual components of the node network stack. The general functionality of the node remains the same in this case. An example would be the replacement of a Network layer component implementing the AODV routing protocol with another implementing the DSR protocol or switching from a single carrier Physical layer component to one using OFDM.

Application level reconfiguration describes the replacement of the entire node network stack with a completely different configuration which carries out a very different function to that of the original set. This level of reconfiguration would apply to a node which switches from operation as a member of an ad-hoc network of UWB devices to a subscriber station within an 802.16 network.

In the implementation of the Reconfigurable Node, a number of features combine to provide each of the degrees of flexibility discussed above. On the level of the individual component, the common component interfaces discussed in

section 3.1 are key to enabling both parametric and structural reconfiguration of the node. The first of these is the Parametric Reconfiguration Interface (PRI). Using the PRI, the parameters of individual components can be reconfigured by specifying the parameter label and providing an associated value. Although this interface is common to each component, the reconfigurable parameters are component-specific so a process involving metadata and automatic code generation is used to provide the specific functionality required at each. A dedicated script uses metadata tags inserted in the component code to generate the reconfiguration code needed. This code is then included in the component implementation. Two additional common component interfaces play vital roles in facilitating structural reconfiguration of the node network stack. The Inter-Component Interface (ICI) forms the link between adjacent components of the network stack. In order to permit alteration of the network stack structure, it is essential that this ICI can be rapidly torn down and re-established. With these requirements in mind, a layer-neutral packet structure was implemented to encapsulate the data flowing between components of the network stack. The use of this packet structure greatly reduces the complexity needed for the ICI and simplifies the process of linking and unlinking components to form the node network stack structure. The Component Lifecycle Interface (CLI) is used to externally control the operation of the individual component and is essential in the process of structural reconfiguration. As the links between components are altered, the CLI is used to suspend and resume component operation in order to avoid potential conflicts, as well as to start the operation of inserted components and stop that of components which are being removed.

Beyond the scope of the individual component, a number of features of the reconfigurable node framework discussed in section 3.2 serve to provide the node flexibility required. Together with the Configuration Parser and the use of XML configuration documents, the Stack Manager performs a central role in node reconfiguration. In order to facilitate any of the levels of reconfiguration discussed above, the SM provides an interface through which configuration documents may be passed. Upon receipt of a new document, the SM uses the CP to compare the new configuration specified with that of the network stack currently in operation. Each difference identified during this process will trigger a specific reconfiguration. The sum total of these reconfigurations will amount to a new network stack state corresponding to the received configuration document.

Figure 4 illustrates the manner in which all of the elements of the Reconfigurable Node discussed above interact to fulfil the roles of observation and action specified by the cognition cycle in section 2.

4. Conclusions

This paper has presented a reconfigurable platform for the development of cognitive networks. The core processes of a cognitive system were discussed and used to define two entities which combine to form a node within a cognitive network - the Reconfigurable Node and the Cognitive Engine. A component-based Reconfigurable Node architecture was presented and the key elements comprising the architecture were examined. Finally, the principal mechanisms of the Reconfigurable Node provided to permit cognitive operation were outlined.

Future work in the areas of opportunistic spectrum access, cross-layer optimization, reconfigurable protocol layers and policy-based reasoning in cognitive networks will be facilitated by the reconfigurable platform.

References

- [1] S. Haykin. Cognitive radio: brain-empowered wireless communications. *Selected Areas in Communications, IEEE Journal on*, 23(2):201–220, 2005. 0733-8716.
- [2] P. Mackenzie. *Software and reconfigurability for software radio systems*. PhD thesis, University of Dublin, Trinity College, 2004.
- [3] J. Mitola III. Cognitive radio for flexible mobile multimedia communications. In *Mobile Multimedia Communications, 1999. (MoMuC '99) 1999 IEEE International Workshop on*, pages 3–10, 1999.
- [4] J. Mitola III and G. Q. Maguire Jr. Cognitive radio: making software radios more personal. *Personal Communications, IEEE [see also IEEE Wireless Communications]*, 6(4):13–18, 1999. 1070-9916.
- [5] C. Szyperski. *Component Software: Beyond Object-Oriented Programming*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2002.
- [6] R. Thomas, L. DaSilva, and A. MacKenzie. Cognitive networks. In *New Frontiers in Dynamic Spectrum Access Networks 2005 (DySPAN), First IEEE International Symposium on*, pages 352–360, 2005.
- [7] <http://xml.apache.org/xerces-c>.
- [8] <http://www.w3.org/XML>.