A Component-based Approach for Constructing Flexible Link-Layer Protocols

André Puschmann*, Mohamed A. Kalil*§, Andreas Mitschele-Thiel*
*Integrated Communication Systems Group, Ilmenau University of Technology, Ilmenau, Germany
§Faculty of Science, Suez University, Suez, Egypt
Email: [andre.puschmann, mohamed.abdrabou, mitsch]@tu-ilmenau.de

***Abstract*—In this paper, we propose a novel approach for constructing flexible link-layer protocols which separate their functionality into multiple components. Our work is motivated by the fact that link-layer protocols are usually implemented as monolithic blocks. This approach limits the possibility of reusing common functions in multiple protocols and also narrows the flexibility of combining existing blocks for creating new protocols. Through flexibly interchangeable components, the proposed architecture addresses these issues. We discuss the design of the architecture and evaluate a prototype that demonstrates cognitive radio functionality such as autonomous link establishment, high-performance spectrum access and self-organized link maintenance after the unexpected arrival of a primary user.**

***Index Terms*—Link-layer protocols, component-based design, CR, SDR**

I. Introduction

The link-layer of a communication system is the protocol layer that is responsible for transferring data between neighbouring nodes. In networks with a shared communication channel the *Medium Access Control* (MAC) sublayer coordinates who gets access to the medium and when. Traditional systems usually assume that the channels are predefined and static and not subject to change without notice.

Within the *Cognitive Radio* (CR) [1] paradigm however, these assumptions do not hold, which introduces new challenges for link-layer protocol research. The main challenge stems from the fact that there is a theoretically unlimited number of channels available for communication. Their availability is likely to change over time due to the mobility of the nodes, changes in the radio environment or through the appearance of a *primary user* (PU) in *Dynamic Spectrum Access* networks. A CR node operating in a licensed frequency band is expected to vacate the occupied spectrum if a PU attempts to use it. This behaviour is referred to as *spectrum mobility* or *handover* and is crucial to avoid causing harmful interference to the original owner of the spectrum. Moreover, opportunistic spectrum access introduces further complexity to the problem of link establishment, i.e. how can two nodes initially find one another, a problem referred to as the *rendezvous problem* [2].

In the recent past, a number of MAC protocols for CR have been proposed by the research community. Most of them are designed as monolithic blocks that tightly integrate either spectrum mobility, rendezvous or both. From a practical perspective, this renders them complex, difficult to implement, and lowers their reusability.

We therefore introduce the concept of a *Flexible Link-Layer* (FLL) that partitions the functions of a link-layer protocol into multiple components. A mediator object is used to control and coordinate the interactions among these components. This has the advantage that protocol components actually can be reused and recombined to new protocols which together may meet other requirements, but *without* reimplementing common functionality from scratch.

To complement the conceptual work presented in this article, we describe a prototype of the proposed approach as well as three representative link-layer configurations that we have developed. Moreover, we evaluate the prototype through two practical experiments within a CR use case. We assess the time to rendezvous as well as the handover delay. The contributions of this article can be summarized as follows:

- Design of a component-based architecture for constructing flexible link-layer protocols with interchangeable and reusable blocks,
- Implementation of a prototype with three link-layer configurations and
- Evaluation of the prototype through practical experiments.

The rest of the paper is organized as follows: Section II discusses related work on MAC and link-layer development for CR. Section III describes the three main link-layer functions addressed in this work. Section IV presents and discusses our proposed link-layer design. Section V describes the implementation of the prototype which is evaluated in section VI. The paper concludes with a description of future work in section VII.

II. Related Work

Component-based development [3] is a design paradigm that originates from software engineering. It aims at designing and developing loosely coupled functional blocks that can be reused and recombined in order to reduce development costs and to improve the overall system quality. The application of component-based techniques to the design of communication system is mainly motivated by the desire to support flexibility and runtime reconfigurability in such systems. Especially the success of wireless sensor networks has led to the development of a large number of MAC and routing protocols for various hardware platforms. The incompatibility and limited interoperability among different solutions has motivated researchers

CROWNCOM 2013, July 08-10, Washington DC, United States
Copyright © 2013 ICST
DOI 10.4108/icst.crowncom.2013.252057

to propose common abstraction layers that hide details of the underlying hardware. The *Sensornet Protocol* for examples, proposed by *Polastre et al.* in [4], provides a unified neighbour management and message pool and allows to implement different link and networking protocols for sensor networks. *Klues et al.* have further developed their idea and proposed a *MAC Layer Architecture* (MLA) with power management support in [5]. The architecture consists of reusable components that implement common MAC features as well as low-level components that abstract details of the underlying sensor node hardware.

A similar approach has been taken by *Ansari et al.* in [6] for the development of MAC protocols using a hardware-software co-design approach based on FPGA boards. The main advantage of FPGA-based MAC designs is their high performance. However, ease of development, which is extremely important for the design of novel experimental and research oriented systems, is usually sacrificed. Moreover, the integration and connection to other components of a CR architecture, such as an environment database, becomes complicated.

Software-based solutions offer greater flexibility but suffer from the disadvantages of digital signal processing on general purpose hardware and high communication delays to and from the radio hardware. Noteworthy work towards an experimental platform based on *GNU Radio* has been presented by *Yang et al.* in [7] and *Nychis et al.* in [8]. In [9], *Mandke et al.* describe a flexible wireless network testbed for MAC and PHY layer research called *Hydra*. Though not explicitly targeted for CR applications, the architecture could be used to implement reconfigurable link-layer protocols using *Click*.

As has been shown by previous research in the field of sensor networks and to some extent also in MAC development, the benefits associated with component-based techniques are favourable for the design of flexible communication systems. A possible reason why this methodology hasn't been applied to the field of CR link-layer protocols is the actual lack of practical implementations in that domain.

III. Link-Layer Functions

Besides providing traditional medium access control, the link-layer of a CR node has to consider aspects which are unique to this technology and not present in conventional communication systems. Moreover, ad-hoc networks introduce further challenges due to the absence of a supporting network infrastructure. This paragraph gives an brief overview about the three main link-layer functions required to control the life cycle of a CR node:

- *Link Establishment:* Link establishment which is often also called rendezvous is usually required when network nodes are initially turned on, or leave the service area covered by a network. In infrastructure-based environments this is often achieved through a dedicated control channel. In ad-hoc networks, however, a centralized control entity is often impractical. Therefore, nodes must be able to establish links among themselves on their own using a rendezvous protocol.
- *Link Access:* Access strategies are inevitable whenever multiple nodes in a network wish to communicate over a shared medium. A MAC protocol coordinates when, which and for what amount of time a certain terminal can access the medium.
- *Link Management:* One of the main challenges for CRs is to peacefully coexist with other devices, particularly with PUs. Therefore, if a CR accesses a licensed channel and the PU (re-)occupies the same channel, the CR is required to vacate the band in order to protect the licensed communication. This function is usually controlled through a spectrum mobility protocol.

IV. Flexible Link-Layer Architecture

CR MAC protocols are usually implemented as monolithic protocols that tightly integrate spectrum mobility, rendezvous or both into the core of their design. This approach limits the possibility of reusing common functions in multiple protocols and also narrows the flexibility of combining existing blocks for creating new protocols. From a practical perspective, monolithic designs are also complex and difficult to implement and maintain.

We therefore introduce the concept of *Flexible Link-Layer* (FLL) and propose to partition the functions of a link-layer protocol into multiple components. Based on the functions defined in Section III, the proposed link-layer architecture is based on three pillars (see Figure 1). Each pillar may be represented by one or more protocol components. Additionally, the *Link-Layer Controller* (LLC) connects the components together and acts as a mediator between the protocols.

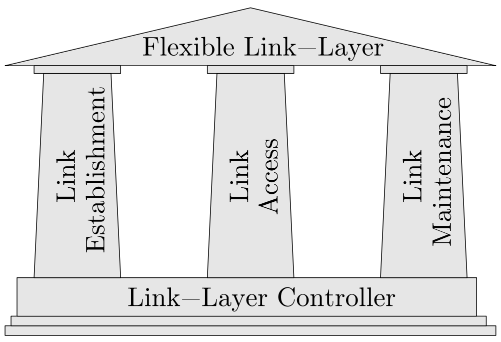

Fig. 1. The three pillars of the flexible link-layer architecture.

A. Design Rationale

FLL aims at simplifying the design, the implementation and the evaluation of link-layer architectures for future wireless communication systems. Rather than addressing all requirements of a given application scenario inside a single protocol, it divides the functions into separate blocks, which, connected through well-defined interfaces, cater for them.

The actual behaviour and the logic of the link-layer is modelled inside the LLC through a *mediator object* which is responsible for controlling and coordinating the interactions among the protocols. The mediator pattern [10] is a design paradigm from software engineering that avoids unnecessary coupling and dependencies and therefore facilitates component reuse.

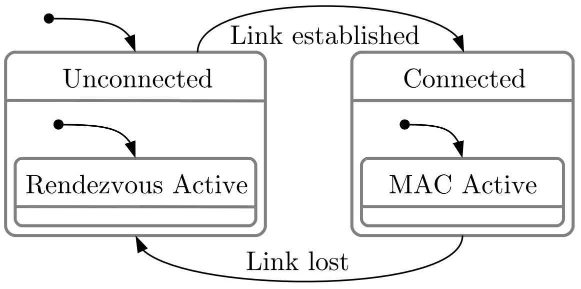

Fig. 2. Example state machine executed inside the LLC.

The interaction within and among components and the transitions between them are triggered through events which may be sent out from multiple sources such as (1) other components of the link-layer, e.g. after an action such as a radio reconfiguration has been completed, (2) other components of the software radio, e.g. a spectrum sensing component which has detected an active PU or (3) any other component in the system, e.g. a user application which reports a change of the experienced quality of service.

The main advantages of the component separation can be summarized as follows:

- *Reusability:* Components which are often needed can be reused in multiple systems rather than developed from scratch every time. The *Carrier Sense Multiple Access* (CSMA) random access scheme for instance is the basis for many CR MAC protocols. Thus, a block that implements such a protocol should be reused in multiple systems.
- *Flexibility:* Reusable blocks can be combined in a flexible manner to build new protocols or to adapt existing ones. To give an example, an existing protocol comprising a MAC component based on static channels could be complemented, without changes to the MAC, by a rendezvous block which establishes links between network nodes if the radio would need to operate in dynamic scenarios. On the other hand, a CSMA-based MAC (which is known to suffer from poor performance in overload situations) could be replaced by a TDMA-based protocol during runtime if the channel congestion exceeds a certain threshold.

B. System Example

To support the understanding of the flexible link-layer concept and to demonstrate the interaction among multiple components consider the following example: imagine a basic MAC protocol has been successfully implemented that operates on a statically configured radio channel. If this system was to be extended to employ the rendezvous concept to operate on a dynamically changing set of channels, the protocol designer would have two options to choose. Either integrating the rendezvous functions into the existing MAC protocol or developing an separate rendezvous component that interacts with the existing MAC.

Figure 2 depicts a simple mediator object that is modelled using a *Finite State Machine* (FSM). The FSM has only two states, either it is connected to another node or it is not. Note that the two main states are compound states that activate/deactivate the corresponding protocol components in their substates.

If a node is turned on, the rendezvous protocol runs until a link could be established successfully. During normal operation, the MAC protocol coordinates the packet exchange between nodes. It gives the control back to the rendezvous block if the link gets lost, due to PU activity for example. In this case the transitions between the states of the FSM are triggered through an internal event issued by the rendezvous block as well as an external event issued by the spectrum sensing block.

The advantage of following the FLL approach in this example is that both the MAC and rendezvous protocol can be reused in other system configurations without reimplementing them.

TABLE I
LINK-LAYER CONFIGURATIONS AND COMPONENT REUSE.

Component \ Configuration	1	2	3
Stop-and-Wait MAC (Aloha)	×		
CSMA MAC		×	×
Random Rendezvous	×		×
Hybrid Spectrum Mobility		×	×

V. IMPLEMENTATION

In this section, we develop an experimental prototype to showcase the applicability and feasibility of the proposed architecture. In total, we have implemented three link-layer configurations with a different range of functions and level of complexity. The configurations are listed in table I. All components were implemented only once and were reused in all other system configurations.

In the remaining part of this article, we will concentrate on link-layer configuration three which is the most complex system. It extends the example described in section IV-B by a *Mobility* component. The demonstrator has been implemented on the basis of Iris [11], [12], an open-source SDR framework. However, the concept can also be realised with *GNU Radio* or any other reconfigurable radio platform.

We now describe each component of the example link-layer configuration in a separate paragraph.

A. Link-Layer Controller

Figure 3 shows the mediator object - again implemented as a FSM - that models the interactions between the protocol components. After powering on, the radio starts in the *Unconnected* state and tries to set up a connection with another node. The two main states are again compound states that activate/deactivate the corresponding protocol components in their substates. In this example the *Rendezvous* protocol remains active until it establishes a link to another node which causes a transition to the *Connected* state.

The connected state is also a compound state with two orthogonal regions which are executed in parallel (see the

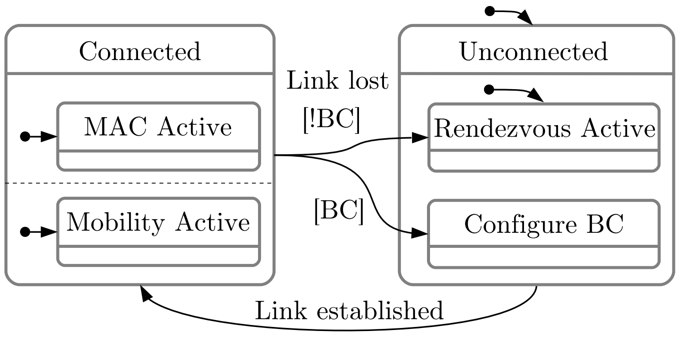

Fig. 3. Finite state machine implemented inside the link-layer controller

dashed line). This means that both, *MAC* and *Mobility* are active at the same time. The spectrum mobility protocol now tries to negotiate a *Backup Channel* (BC) with its peers while the MAC can be used for normal communication.

The radio is constantly monitoring the spectrum in order to detect any PU activity in its surrounding. The spectrum sensing component then issues a *Link lost* event which causes the FSM to transit into the unconnected state again. Whether the rendezvous protocol is activated again or the radio tunes to the BC depends on the result of the negotiation process.

In this configuration, the radio reconfiguration is only taking place in the *Unconnected* state either while hopping over the set of available channels during rendezvous or to tune to the BC.

B. Rendezvous

If the radio is initially turned on or if connectivity is interrupted, a rendezvous protocol is required to establish links among the nodes of a network. We have implemented a rendezvous component for Iris and the corresponding beaconing protocol that is agnostic to the actual sequence generation algorithm. In this example, we are using a random algorithm [2] in which a node that wishes to join a network randomly hops over the set of available communication channels m_i until rendezvous occurs. In each time slot, each node selects a channel c with probability $1/m_i$. This procedure runs until two nodes have selected the same channel and one of the nodes has received a rendezvous beacon transmitted by the other one. If a third node wishes to join the network, the same protocol can be used as well.

C. Medium Access Control

During normal operation of the radio, the MAC protocol is responsible for coordinating the access to the channel shared by multiple users in the network. The MAC component is only allowed to access the channel in this state. It is inactive (i.e. does not produce any output data) in all other states. All outgoing packets are buffered inside the MAC until the node is connected again.

The actual MAC protocol is implemented as a *Stack component* inside the Iris framework. In this example a CSMA-based protocol is used [13].

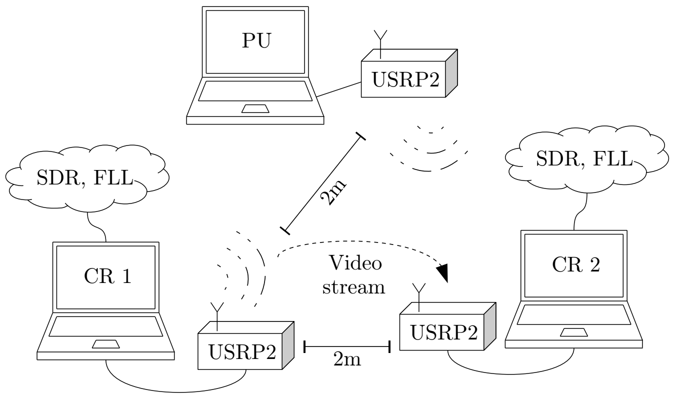

Fig. 4. Experimental system setup: two CR nodes and one PU.

D. Spectrum Mobility

The main task of the *Spectrum Mobility* component is to maintain network connectivity and to guide the behaviour of the radio in case the current operating channel is no longer available for secondary use. This function is crucial to avoid causing harmful interference to the PU as well as to minimize performance degradation during the spectrum handover procedure.

In this example, we have realized a *hybrid* spectrum mobility strategy according to the classification suggested by *Christian et al.* in [14]. We have combined proactive spectrum sensing and negotiation with reactive spectrum handoff. In other words, the negotiation process inside the *Mobility Active* state is executed *before* the arrival of the PU. However, the actual reconfiguration of the radio inside the *Configure BC* state is carried out *after* the detection of the PU. To negotiate a backup channel between a pair of nodes, we have implemented an algorithm which first requires that all nodes broadcast a control beacon including the set of available channels. A *master* node is then selected based on the node identification. The master then determines a common BC among all nodes and informs them about the decision. The result of the negotiation process is stored inside a the LLC which maintains the current operating channel as well as the backup channel for its network cluster. The control packets are transmitted in-band along with ordinary data packets.

VI. Evaluation

In this section, we evaluate the prototype implementation through practical experiments. Thereby, we measure the time to rendezvous and the handover delay. Furthermore, we describe the component reuse across the three link-layer configurations.

A. Experiment Setup and Description

The experiment consists of two CR nodes operating in the 2.4 GHz ISM band. Both are running the prototype implementation with link-layer configuration three described in the previous section. Each node is equipped with one USRP2 device. The spectrum is divided in 25 adjacent channels with a bandwidth of 2 MHz each ($2400 + i \cdot 2$ MHz with $i \in [0..24]$).

TABLE II
HARDWARE CONFIGURATION AND SYSTEM PARAMETERS

PARAMETER	VALUE
RF hardware	USRP2 and XCVR2450
Sampling rate	2 Msamples/s
No. of channels	5 to 25
f_{center}	$2.401 - 2.451$ GHz
Channel bandwidth	2 MHz
Modulation scheme	QPSK
Subcarriers (data)	64 (44)
Rendezvous slot length	500 ms

Both nodes observe the same set of channels and have the same naming structure which corresponds to the *shared system model* [2].

A third node acts as a PU broadcasting a pseudo-random OFDM signal and arbitrarily switching between one of the available channels.

The nodes implement the behaviour described in the previous section. If they are initially powered on, they immediately attempt to rendezvous with one another in one of the available channels. As soon as they are connected, a common BC is negotiated.

A video streaming application running on both nodes is transmitting from one node to another. In case they detect an active PU on the current operating channel, the MAC immediately suspends any active transmission and transits into the unconnected state in order to protect the licensed user from harmful interference.

The LLC now tunes the radio to the negotiated backup channel. After that, the MAC protocol can resume its ongoing communication.

Note that in this experiment, we assume a similar radio environment at both, the sending and the receiving node. Both nodes are within sensing range of the PU and are therefore able to detect the transmission. However, they perform the sensing and spectrum handover autonomously and are not supported by a central coordinator or common control channel.

The hardware configuration as well as the parameter settings of the radio are summarized in table II.

B. Time To Rendezvous

Time To Rendezvous (TTR) is the main metric to assess the performance of a rendezvous algorithm. Typically expressed in slots, it is the amount of time it takes for two or more nodes to initially find one another and set up a link.

In our experiments, we configured the slot length to be 500 ms. In each slot, the radio was tuned to a randomly selected channel and *one* rendezvous request beacon was sent. Upon receiving a beacon, a node immediately returns a rendezvous reply packet and the procedure is stopped.

We have set the number of operating channels to 5, 10, 15, 20 and 25 and repeated the experiment 15 times for each configuration.

The results of this experiment are plotted in figure 5. As one can see from the graph, the mean number of slots required for successful rendezvous varied between 5 and 15 with a total number of channels of 5 and 25, respectively. This fits well with the results obtained through a Python simulation with 100.000 runs.

A similar experiment has been carried out in [15] with a set of four USRP nodes and GNU Radio. The conclusion that has been drawn was that rendezvous was never successful without sending multiple beacons within one slot, no matter how many channels or which transmit probability was used. During our work we never experienced such a behaviour. A possible explanation for this could be a high radio reconfiguration delay resulting in a *blind phase* of the radio, effectively missing beacons during that time.

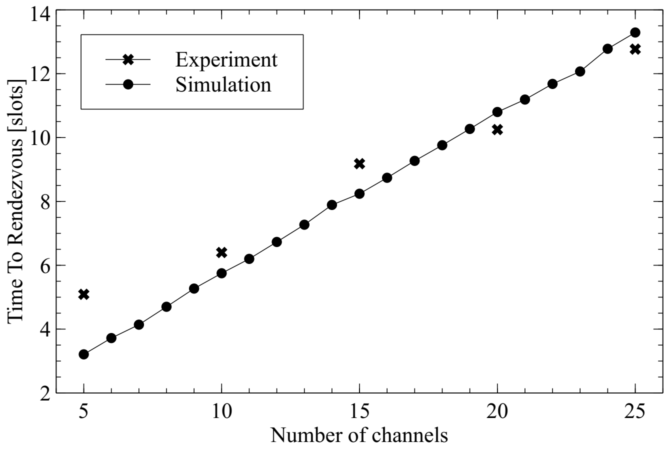

Fig. 5. TTR vs. number of channels using random rendezvous.

C. Handover Delay

In this section, we study the handover delay of the system with varying spectrum sensing intervals. The handover delay is defined as the time the system needs to recover from an active PU. This period includes the time needed to detect the PU activity, suspend the own transmission, query and tune to a new channel and resume the transmission at the sender and the receiver. The delay is measured at the receiver node by calculating the data throughput every 100 ms. We have varied the sensing interval between 500 ms and 1 s and repeated the experiment 25 times for each interval.

The results of the experiment are shown in table III. We observed that the minimum handover delay is almost the same for all sensing intervals. Assuming that in this case the PU activity could be detected successfully shortly after it has been activated, 300 ms can be seen as the minimum reaction time. While the minimum value is almost constant for all interval times, the average value clearly indicates the positive impact of a shorter sensing interval. In all cases, the maximum handover delay was larger than twice the actual interval time. This is because of a non-zero probability of misdetection at either the transmitter or receiver. In such a case, two or more cycles were needed to successfully detect the presence of the PU. The average handover delay obtained during the experiments is significantly lower than similar measurements done by *Denkovski et al.* in [16]. The authors reported a handover duration of around 1.3 s which is mainly needed

TABLE III
EXPERIENCED HANDOVER DELAY IN SECONDS USING DIFFERENT SENSING INTERVALS, AVERAGED OVER 25 RUNS

INTERVAL	MIN	AVG	MAX	SD
1000 ms	0.4	1.31	3.4	0.86
750 ms	0.4	0.89	1.8	0.47
500 ms	0.3	0.74	1.5	0.35

for component reconfiguration. In their work, however, the channel handover was trigged by a change to the radio access policy.

D. Component reuse

Even though flexibility and modularity are difficult to measure, they are best shown by looking at a practical example that benefits from the component reuse feature. We have implemented three different link-layer configurations which are built out of four different protocols. The configurations of all three systems are listed in table I. All components could be reused without further modification and only the mediators needed to be implemented. In total we've implemented four components plus three mediators which sums up to seven *blocks*.

By using a conventional protocol engineering approach, we would have needed to reimplement blocks for each system although they were already present. This would have created lots of redundant code and would have complicated any modification to an existing protocol. Without the provided concept, we would have needed to write seven components plus the additional code that implements the interaction among the protocols, a total of ten software blocks. We therefore conclude that the proposed protocol separation reduces the amount of components and source code required to implement, in this specific example by 30%.

VII. CONCLUSION

In the past, most link-layer protocols had to be developed from scratch every time a new feature was needed. With the shift towards a component-based paradigm, the flexible link-layer concept proposed in this article allows to reuse protocols and components across different system configurations. A link-layer configuration is represented through a mediator object which encapsulates the communication and interaction between two or more protocols. This results in an overall lower development time for new protocols and inherently increases their quality. To showcase the feasibility of this approach, we have implemented a prototype for Iris and evaluated its flexibility through the realization of three representative link-layer configurations. Using the last configuration, we have practically demonstrated CR functionality such as autonomous link establishment, high-performance spectrum access and self-organized link maintenance after the unexpected arrival of a PU. Empirical results show that the component-based approach achieves comparable performance to monolithic designs but at the same time reduces the amount of code that needs to be developed. We are currently enhancing the design and experimenting with a new TDMA-based MAC protocol with fixed channel access delays.

ACKNOWLEDGMENT

This work has been carried out within the International Graduate School on Mobile Communications (Mobicom), supported by the German Research Foundation (GRK1487) and the Carl Zeiss Foundation.

REFERENCES

[1] S. Haykin, "Cognitive Radio: Brain-empowered Wireless Communications," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 2, Feb. 2005.

[2] N. Theis, R. Thomas, and L. DaSilva, "Rendezvous for Cognitive Radios," *IEEE Transactions on Mobile Computing*, Feb. 2011.

[3] C. Szyperski, *Component Software: Beyond Object-Oriented Programming*, 2nd ed. Addison-Wesley Professional, Oct. 2002.

[4] J. Polastre, J. Hui, P. Levis, J. Zhao, D. Culler, S. Shenker, and I. Stoica, "A unifying link abstraction for wireless sensor networks," in *ACM Conference on Embedded Networked Sensor Systems (SenSys)*, San Diego, California, Nov. 2005.

[5] K. Klues, G. Hackmann, O. Chipara, and C. Lu, "A Component-Based Architecture for Power-Efficient Media Access Control in Wireless Sensor Networks," in *ACM Conference on Embedded Networked Sensor Systems (SenSys)*, Sydney, Nov. 2007.

[6] J. Ansari, X. Zhang, A. Achtzehn, M. Petrova, and P. Mähönen, "A Flexible MAC Development Framework for Cognitive Radio Systems," in *IEEE Wireless Communications and Networking Conference (WCNC)*, Mar. 2011.

[7] L. Yang, Z. Zhang, W. Hou, B. Y. Zhao, and H. Zheng, "Papyrus: A Software Platform for Distributed Dynamic Spectrum Sharing Using SDRs," *ACM Computer Communication Review (CCR)*, Jan. 2011.

[8] G. Nychis, T. Hottelier, Z. Yang, S. Seshan, and P. Steenkiste, "Enabling MAC Protocol Implementations on Software-defined Radios," in *Proceedings of the 6th USENIX Symposium on Networked Systems Design and Implementation*, Apr. 2009.

[9] K. Mandke, S.-H. Choi, G. Kim, R. Grant, R. Daniels, W. Kim, R. Heath, and S. Nettles, "Early Results on Hydra: A Flexible MAC/PHY Multihop Testbed," in *IEEE 65th Vehicular Technology Conference (VTC)*, Apr. 2007.

[10] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1994.

[11] P. D. Sutton, J. Lotze, H. Lahlou, S. A. Fahmy, K. E. Nolan, B. zgl, T. W. Rondeau, J. Noguera, and L. E. Doyle, "Iris: An Architecture for Cognitive Radio Networking Testbeds," in *IEEE Communications Magazine*, vol. 48, no. 9, Sep. 2010, pp. 114–122.

[12] Software Radio Systems Ltd., "The Iris project page," available under http://www.softwareradiosystems.com/redmine/projects/iris; last visited on Apr. 15, 2013.

[13] A. Puschmann, M. A. Kalil, and A. Mitschele-Thiel, "A Flexible CSMA based MAC Protocol for Software Defined Radios," *Frequenz Journal of RF-Engineering and Telecommunications*, vol. 6, Oct. 2012.

[14] I. Christian, S. Moh, I. Chung, and J. Lee, "Spectrum Mobility in Cognitive Radio Networks," *IEEE Communications Magazine*, vol. 50, no. 6, pp. 114–121, Jun. 2012.

[15] M. D. Silvius, A. B. MacKenzie, and C. W. Bostian, "Rendezvous MAC Protocols for Use in Cognitive Radio Networks," in *IEEE Military Communications Conference (MILCOM)*, Oct. 2009.

[16] D. Denkovski, V. Pavlovska, V. Atanasovski, and L. Gavrilovska, "Novel Policy Reasoning Architecture for Cognitive Radio Environments," in *IEEE Global Telecommunications Conference*, Miami, USA, Dec. 2010.